English

Fujitsu Software BS2000

BS2ZIP

ZIP Archiving in BS2000

User Guide

Valid for:
BS2ZIP V21.0B33

Edition November 2025

Comments… Suggestions… Corrections…

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to: bs2000.info@fujitsu.com.

Certified documentation according to DIN EN ISO 9001:2015

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2015.

Copyright and Trademarks

Table of Contents

BS2ZIP V21.0B

1 Introduction

With the TU program BS2ZIP, ZIP archives can be created in BS2000 systems and files can be added to or extracted from these archives. A BS2ZIP archive can be created in BS2000 or Windows-compatible format. BS2ZIP has a simple and easy-to-use interface for SDF statements and a programming interface.

BS2ZIP is part of the BS2000 OS DX operating system package.

1.1 Objectives and target groups of this manual

This manual is intended both for privileged and nonprivileged BS2000 users.

1.2 Summary of contents

BS2ZIP provides the following functions:

- The creation of a BS2ZIP container compliant with WinZip® (PKZIP 4.5)
- The addition of all kinds of BS2000 files and organization: PAM, ISAM, SAM and PLAM (separate library elements also being possible) using the compliant compression method “Zlib”: RFC 1950 Copyright ©1995-2003 Jean-Loup Gailly and Mark Adler: jloup@gzip.org / madler@alumni.caltech.edu
- The extraction and restore of the BS2000 files with their original DMS attributes
- The display of the available files included in the BS2ZIP container
- The support of multiple files in the BS2ZIP container
- Support of standard Zip 2.0 encryption when files are added or extracted
- The interoperability with Windows environment:
  - Under BS2000:
    Opening of WinZip made transferred archive files in BS2000 as BS2ZIP container (transfer via ftp)
    Extracting files as text files (SAM files) with automatic detection of encoding between ISO8859F/WCP1252 /UTF8/UTF16 and conversion of ISO8859F and WCP1252 to EBCDIC by default or conversion from a manually set encoding.
    Extracting files as raw files (binary) on request (PAM files)
  - Under Windows:
    Opening of transferred BS2ZIP container using WinZip (transfer via ftp)
    Conversion of EBCDIC text files (such as SAM/ISAM) ensured by the application (conversion EDF0x into ISO8859-15/WINANSI).
- Interoperability with Linux/Unix system environment.
  The possibility of reading and extracting files from GZIP archives.

A program interface in C++ is provided that offers all the functionalities of the TU application.

Additional product information

Current information, version and hardware dependencies, and instructions for installing and using a product version are contained in the associated Release Note. These Release Notes are available online at http://bs2manuals.ts.fujitsu.com.

1.3 Changes since the last edition of the manual

The following changes have been implemented for BS2ZIP V21.0A20:

EXTRACT-FILE statement:

With operand CHARACTER-CONVERSION it is possible to define a source and target coded character set that is used for conversion of SAM or ISAM files during extraction.

Several commands were extended with parameter *PATH-NAME for specifying Unix/Windows compatible file names without interpreting special characters (* or /) as BS2000 wildcards. These commands with their parameters are ADD-FILE TO-FILE, DELETE-FILE FILE-NAME, EXTRACT-FILE FILE-NAME, SHOW-FILE-ATTRIBUTES FILE-NAME.

SHOW-FILE-ATTRIBUTES will show unavailable attributes of Gzip files as NOT AVAILABLE.

The new S variable LAST-BYTE-VALID was added to structured output of SHOW-FILE-ATTRIBUTES command.

Minor changes were done to the descriptions of command ADD-FILE and its parameters TO-FILE and DATA-TYPE to clarify behavior of the program.

The following changes have been implemented for BS2ZIP V21.0B00:

OPEN-ZIP-CONTAINER and CONVERT-ZIP-CONTAINER statement:

Newly created archives, as well as those converted from SAM to PAM, will have the Last Byte Pointer enabled by default. If the Last Byte Pointer is not desired, the user can apply an optional rep that reverses this change.

The following changes have been implemented for BS2ZIP V21.0B10:

ADD-FILE, EXTRACT-FILE, SHOW-FILE-ATTRIBUTES statement:

Now during ADD-FILE BS2ZIP stores additional metadata about files in archive not only inside of the file-info, but also inside of a extensible data field with a tag 0xBC6E. Some of the additional metadata fields will be output by SHOW-FILE-ATTRIBUTES command and it will be used to restore the original attributes of the file more accurately during EXTRACT-FILE. Also both sub operands of EXTRACT-FILE now accept any CCSNAME, which is defined in XHCS, while TO-CCS sub operand can accept a default value of *STD.

EXTRACT-FILE statement:

Before extracting files from archive, that were added there on an open systems, a logic that determines the most likely encoding of the member runs if CHARACTER-CONVERSION is not set to *BY-PARAMETERS, *TO-WIN-ANSI or *TO-EBCDIC. This logic picks from the list of EDF04F, ISO8859-F, UTF-8, UTF-16. If *TO-WIN-ANSI is specified, then current encoding is assumed to be EDF04F. If *TO-EBCDIC is specified, then current encoding is assumed to be ISO8859-F.

The following changes have been implemented for BS2ZIP V21.0B20:

For ADD-FILE the new operand DELIMITER is introduced. With this operand the user can control the usage of line delimiters of files added to a winzip compatible archive. This feature is useful in case the files are extracted later on Linux or other *nix systems.

If files from open systems platforms are extracted from a winzip compatible archive overwriting an existing file the specified CODED-CHARACTER-SET of the existing file specifies the desired output encoding.

The following changes have been implemented for BS2ZIP V21.0B30:

For EXTRACT-FILE the new operand DELIMITER is introduced. It allows to specify delimiter used based on the type of encoding that the file stored in the archive. Also it is possible to specify the binary representation of the delimiter explicitly.

Empty strings of text files are no longer padded with a space character during EXTRACT-FILE. But it is possible to restore this behavior with PAD-EMPTY-RECORD = * YES.

Standard conversion is the default behavior for character conversion.

The following changes have been implemented for BS2ZIP V21.0B31:

Encryption and compression is inapplicable to empty file as they are stored without compression as they only require to store their metadata without initialization vectors.

The following changes have been implemented for BS2ZIP V21.0B32 an V21.0B33:

New CCSNAME type Unknown - for handling files that have a CODED-CHARACTER-SET that cannot be interpreted as EBCDIC, ASCII or Unicode by both BS2ZIP and XHCS.

SHOW-FILE-ATTRIBUTES displays not only current and original CCSNAME, but also their CCS types for files added to archives by BS2ZIP.

1.4 Notational conventions

The following typographical elements are used in this manual:

¡	For notes on particularly important information
!	This symbol designates special information that points out the possibility that data can be lost or that other serious damage may occur.

[ ]	References to other publications within the text are given in abbreviated form followed by numbers; the full titles are listed in the “References” section at the back of this manual.
`input`	Inputs and system outputs in examples are shown in typewriter font

2 BS2ZIP application overview

2.1 ZIP containers

The BS2ZIP application enables ZIP containers to be created on BS2000 systems, and files to be added to and extracted from these containers. The ZIP compression used is the Zlib method.

This application is also able to read ZIP containers built on open systems (Windows, Unix systems, Linux) provided they use the Zlib compression method and provided they are compliant with PKZIP 4.5 or GZIP.

The containers created in BS2000 can also be reopened on open systems if they were created in WinZip compatible format.

The command to start the BS2ZIP application is /START-ZIP-MANAGER or /START-ZIP.

After starting the application, the user has to open an existing container or to create a new one with the OPEN-ZIP-CONTAINER statement. In the case of the creation of a new container the user has to select the appropriate container format (operand FORMAT of the statement) according to the usage planned for the container to be created.

Please respect the following rules to select the appropriate format:

1. **Container creation** :
   - if you plan to use the container on BS2000 platforms only, you can create it in BS2000 format. Also, if you zip SAM or ISAM file including special print control characters, use the BS2000 format.
   - if you plan to export the container on open systems, create your container in *WINZIP compatible format (default one) only.
2. **Container open** :
   - you don’ t need to specify the container format at open of an existing container. The program finds itself the container format. However, if you specify the format, it must be the real format of the container otherwise the OPEN-ZIP-CONTAINER statement is rejected. Once, the container is opened, you can add or extract files.

The following table summarizes the default behavior of the BS2ZIP application for the different file types according to the container open format:

ZIP Format	File Access					
	SAM		ISAM		PAM/PLAM	
WINZIP compatible	A:	Data only are saved WIN-ANSI conversion if file is EBCDIC encoded and not a variant of Unicode (e.g. UTFE); CRLF (000D000A for UTF16, 0D25 for EBCDIC, 0D0A for the rest) are inserted as record delimiters	A:	Data and keys are saved WIN-ANSI conversion if file is EBCDIC encoded and not a variant of Unicode (e.g. UTFE); CRLF (000D000A for UTF16, 0D25 for EBCDIC, 0D0A for the rest) are inserted as record delimiters	A:	Binary save without any conversion

	X:	Files are extracted with same organization than the original file. EBCDIC conversion if necessary Record rebuild according to delimiter saved in the extensible data field or corresponding to encoding if not available *Default processing if no organization info is associated to the file*	X:	Files are extracted with same organization than the original file. EBCDIC conversion if necessary Record rebuild according to delimiter saved in the extensible data field or corresponding to encoding if not available	X:	Files are extracted with same organization than the original file. Binary rebuild
BS2000	A:	Record length saved with data No conversion	A:	Record length saved with data No conversion	A:	Binary save
	X:	Files are extracted with same organization than the original file. No conversion	X:	Files are extracted with same organization than the original file. No conversion	X:	Files are extracted with same organization than the original file. Binary rebuild

A: Add
X: Extract

Using the ADD-FILE and EXTRACT-FILE statements you can modify this default behavior. See those statements for details.

Note

Here and everywhere else in this document, where UTF16 is mentioned, UTF16 big endian is implied. Little endian variant of UTF16 is not supported as it is not supported by XHCS. When files in UTF16 are added/extracted to/from archives, BS2ZIP does not add BOM if it is absent in the file and it does not remove it if it present.

There are 4 types of CCSNAMEs in BS2ZIP: *EBCDIC, *ASCII, *Unicode, *UNKNOWN (see table below). ADD-FILE and EXTRACT-FILE statements have the CHARACTER-CONVERSION option to explicitly specify what type of conversion is required. During execution of the ADD-FILE statement BS2ZIP preserves CCSNAME of an original file by putting it into the file-info (file comment that preserves the original data attributes), and it saves the current encoding of the file inside of the extensible data field. It helps EXTRACT-FILE to use CHARACTER-CONVERSION to determine what kind of conversion is needed to restore the file. By default, EXTRACT-FILE restores the original file encoding; but CHARACTER-CONVERSION also allows to force the output file to be in ASCII or EBCDIC or to force no conversion during the extraction. Unicode types are not converted during ADD-FILE or EXTRACT-FILE unless CHARACTER-CONVERSION = *BY-PARAMETERS is used. Files with unknown type are prohibited from character conversion and they can be added only with CHARARCTER-CONVERSION = *NO; these files can be extracted only with CHARARCTER-CONVERSION = *NO or CHARARCTER-CONVERSION = *BY-PARAMETERS to ignore the metadata of the files and specify the current encoding and desired encoding of the output files.

***EBCDIC**	***ASCII**	***UNICODE**	***UNKNOWN**

EDF03IRV	ISO88591	UTF8	All CCSNAMEs that
EDF03DRV	ISO88592	UTFE	are not in this table
EDF04DRV	ISO88593	UTF16	and that XHCS could
EDF041	ISO88594		not identify
EDF042	ISO88595		
EDF043	ISO88597		
EDF044	ISO88599		
EDF045	ISO8859F		
EDF046	WCP1252P		
EDF047			
EDF049			
EDF04A			
EDF04B			
EDF04C			
EDF04D			
EDF04E			
EDF04F			
EEHCL2			
EEHCLC			
EEHCLC1			
EEHCLAA			
EEHCLG			

Files added by BS2ZIP always have comments starting with "*BS2:" and containing the original BS2000 file attributes. If this comment is missing, BS2ZIP assumes that the file comes from a non-BS2000 system. By default, BS2ZIP now scans the first 32 kiB of the file before extracting to determine its encoding (it distinguishes between ISO8859F, WCP1252, UTF8 and UTF16). To suppress this step, use the operand CHARACTER-CONVERSION=*TO-EBCDIC (file is converted from ISO8859F to EDF04F) / *TO-WIN-ANSI (file is converted from EDF04F to ISO8859F) / *BY-PARAMETERS (input and output encodings are specified manually).

Alternatively, operand WRITE-MODE allows to skip this scan for the extracted foreign text file. If WRITE-MODE = *REPLACE-ONLY or WRITE-MODE = *ANY is specified, then BS2ZIP can write the extracted file from archive to an existing file preserving its CODED-CHARACTER-SET attribute. Type of character conversion is inferred from this CODED-CHARACTER-SET attribute and CHARACTER-CONVERSION operand value.

In the case of file added to WinZip compatible container prior to version V21.0B10 CRLF (0D0A) always indicates the end of a data record, while starting from version V21.0B10 line delimiter is picked in accordance to the encoding of the file (000D000A for UTF16, 0D25 for EBCDIC, 0D0A for the rest). The original file may contain 0D0A or 0A represented in corresponding encoding, otherwise the file is spoiled when it is extracted. A WinZip compatible container should not contain any non-printable SAM or ISAM files.

ADD-FILE

CHARACTER CONVERSION	Meaning
*BY-CONTAINER-FORMAT	Default value. Behavior determined by the ZIP container open format
*NO	No conversion is forced
*TO-EBCDIC	EBCDIC conversion is performed (only for SAM/ISAM files in ASCII encoding)
*TO-WIN-ANSI	WIN-ANSI conversion is performed (only for SAM/ISAM files in EBCDIC encoding)

EXTRACT-FILE

The open format determined by ZIP container format can be modified at EXTRACT-FILE time:

DATA-TYPE	Container format	
	WINZIP-COMPATIBLE	BS2000
*NOT-SPECIFIED	If no file-info found in ZIP container Then *CHARACTER is assumed Else file-info is used - EBCDIC conversion done according to file type	If no file-info found Then error Else file-info is used - no conversion
*CHARACTER	File is extracted as a SAM file Records are delimited by CRLF or LF EBCDIC conversion performed	Statement rejected
*BINARY	File is extracted as a PAM file No conversion performed	Statement rejected
*SAM-BINARY	File is extracted as a SAM file, REC-FORM=U No conversion performed	Statement rejected

CHARACTER-CONVERSION	Meaning
*BY-CONTAINER-FORMAT	Default value. Behavior determined by the ZIP container open format
*NO	No conversion is forced

TO-WIN-ANSI	WIN-ANSI conversion is forced (only for extraction to SAM/ISAM file)
*TO-EBCDIC	EBCDIC conversion is forced (only for extraction to SAM/ISAM file)
*BY-PARAMETERS (...)	Conversion will be made according to specified source and target character sets (only for extraction to SAM/ISAM file)
FROM-CCS	Origin coded character set, from which all characters will be converted.
TO-CCS=*STD	Target coded character set is picked by finding the first fully compatible coded character set in EBCDIC of the same ISO code variant number as origin coded character set. If origin coded character set is Unicode, than no conversion is performed. (ISO code variant number, see manual XHCS [5]).
TO-CCS	Target coded character set, to which all characters will be converted.

Block format when extracting to various disk formats

When extracting a file, BS2ZIP takes into account the disk format in which the file is stored, the block format (BLOCK-CONTROL-INFO) of the original file which is stored in the archive, and the specification in the BLOCK-CONTROL-INFO operand which controls whether the block format must be retained. As a result of this, in some cases the file cannot be extracted, or a block format which differs from that of the original file must be selected.

Extracting to K disks

The extracted files contain the block format of the original file.

Extracting to NK2 disks

Files with the PAMKEY block format cannot be extracted using the BLOCK-CONTROL= *KEEP option.

Depending on the archive format, the BLOCK-CONTROL=*IGNORE option is used to set the block format as follows:

Properties of the original file		**Specification in the BLOCK-CONTROL operand**			
		(BS2000 format)		(WINZIP-COMPATIBLE format)	
FCB type	**BLK-CTRL**	***KEEP**	***IGNORE**	***KEEP**	***IGNORE**
PAM	PAMKEY	Not extracted	NO	Not extracted	NO
PAM	DATA	Like original	Like original	Like original	Like original
PAM	NO	Like original	Like original	Like original	Like original
SAM	PAMKEY	Not extracted	DATA	Not extracted	DATA
SAM	DATA	Like original	Like original	Like original	Like original
ISAM	PAMKEY	Not extracted	DATA2K	Not extracted	DATA2K
ISAM	DATA	Like original	Like original	Like original	Like original

2.2 Transferring files

In BS2000, ZIP containers are always PAM files. They can be transferred to other BS2000 systems or to systems with other operating systems (MS/Windows, Unix, Linux, zOs). In non-BS2000 systems only ZIP containers in WinZip-compatible format can be used.

A ZIP container in PAM file format can be transferred to another system using ftp or openFT V11.0 and higher. Binary mode must be set for the transfer.

A ZIP container in PAM file format can be converted to a SAM file with RECORD-FORMAT=U or vice versa using the //CONVERT-ZIP-CONTAINER statement. In this case the converted file is stored as a copy in the output file.

If no name is specified for these output files, BS2ZIP generates standard file names for these output files as follows:

<name>.SAM[.ZIP] or **<name>.PAM[.ZIP]**

where <name> is a basename with a limited length.

Examples

Name of the ZIP container	File format of the ZIP container	Name of the copy
XYZ	PAM	XYZ.SAM
XYZ	SAM (with RECORD-FORMAT=U)	XYZ.PAM
XYZ.PAM	PAM	XYZ.SAM
XYZ.SAM	SAM (with RECORD-FORMAT=U)	XYZ.PAM
XYZ.ZIP	PAM	XYZ.SAM.ZIP
XYZ.ZIP	SAM (with RECORD-FORMAT=U)	XYZ.PAM.ZIP
XYZ.PAM.ZIP	PAM	XYZ.SAM.ZIP
XYZ.SAM.ZIP	SAM (with RECORD-FORMAT=U)	XYZ.PAM.ZIP

Notes

- When the resultant file name, including catalog and user IDs, exceeds 54 characters, BS2ZIP truncates the basename (XYZ in the example).
- The converted file is created with the userid and the default cat-id of the user, even if the original file is on another userid or catid.

2.3 Encrypting data

When a file is added to a ZIP file, it can be encrypted. This encryption offers such a file in the ZIP container a degree of protection against unauthorized access. Anyone wishing to read the data in the file must decrypt it again when it is extracted and requires the appropriate key (the crypto password) to do this. The ZIP container itself is not protected by this encryption.

BS2ZIP supports the Zip 2.0 encryption mechanism and is consequently compatible with WinZip and many other Zip tools. Zip 2.0 encryption is relatively weak and does not offer sufficient protection against special password recovery tools.

BS2ZIP encrypts files when they are added to the archive and decrypts them again when they are extracted, provided encryption was enabled using the MODIFY-ZIP-OPTIONS statement. The appropriate crypto password must also be specified when the function is enabled.

If a ZIP container contains files with different passwords, encryption must be specified again with the requisite password before a file is extracted.
When extraction takes place to non-BS2000 systems, WinZip or the Zip tool would have to be restarted for each password in such cases.

2.4 File types supported

BS2ZIP supports SAM, ISAM and PAM files including PLAM libraries.

ISAM Files

ISAM files with secondary keys are not supported.

Index and data of the resulting ISAM file after the extract statement are never separated. The resulting file after extraction may be greater than the original one.

#The padding (PADDING-FACTOR) associated to an extracted ISAM file is always the default one (DMS limitation).

PLAM Libraries

BS2ZIP processes PLAM libraries as a whole file.

Library elements can also be processed separately. The file information which is stored for the element in the PLAM library (if none is stored the PLAM default settings are used) is transferred to the ZIP archive and evaluated when extraction takes place.

Temporary Files

BS2ZIP enables temporary files to be added and extracted. These files are recorded under their cataloged file names, i.e. including the tempfile prefix.

As the tempfile prefix is assigned by the system on a task-specific basis, no files with a different tempfile prefix can be created in a task. It is only possible to extract a file with a tempfile prefix without renaming it if the file was added in the same task. Otherwise the file must be renamed explicitly when it is extracted.

Load modules of a PAMKEY disk

Load modules of a PAMKEY disk that have been added to a container cannot be extracted on a NK disk, even if the operand BLOCK-CONTROL-INFO=*IGNORE is set, because the generated NK load module does not contain all the necessary information needed to be loaded correctly on a NK disk.

If the operand BLOCK-CONTROL-INFO=*KEEP is set, you will receive the DMS error message:

```
DMS0D80 ALLOCATED DISK SPACE DOES NOT MATCH WITH THE REQUESTED FILE FORMAT.
```

2.5 File Extents and user-specific resources

It can happen, that the maximum of extents for the ZIP container is reached (310). If the amount of data to compress is important, it is advised to estimate the final size of the container and to open it by link to avoid this. See OPEN-ZIP-CONTAINER “Notes” (OPEN-ZIP-CONTAINER) for details and example.

Ensure that there is enough space in the user ID where the ZIP container is built. If there is not enough space, the container may be incoherent and so unusable. Its content cannot be accessed any more.

If you want to process a large number of files using BS2ZIP, sufficient user address space must be available under your user ID (see ADDRESS-SPACE-LIMIT in the user entry).

2.6 K2 support

Pressing the K2 key has a limited scope. It allows the program to be interrupted in statement mode (//). After an interruption using the K2 key, the user is in command mode (/).

If K2 is pressed during the ADD-FILE, EXTRACT-FILE or SHOW-FILE-ATTRIBUTES statements, BS2ZIP interrupts processing or output with the query message `SZP0208`. Users then have the following options:

- They can simply continue current processing.
- They can abort current processing. When they do this, they return to statement mode (//). The statement may need to be reissued for files which had not been processed by the time the interruption occurred.

2.7 Crash resistance

If the processing of the ADD-FILE statement is abnormally terminated (e.g. in case of system crash or power failure), a zip container may be inconsistent. This inconsistency occurs because the directory containing the information about the container content has been overwritten.

The following mechanism enables the recovery of such corrupt containers: Before the processing of the first ADD-FILE statement, a copy of the container directory and the name of the container are saved in a backup file named as BS2ZIP.YYYY-MM-DD.HHMMSS.BAK. This file is deleted when the BS2ZIP application is normally terminated.

After abnormal termination the file is not deleted. The next time when the inconsistent container is opened, the system automatically searches for the corresponding backup file and restores the container in the state before the start of the last BS2ZIP session. However, the data added during that session still exist in the archive, but are not accessible. The repaired container is therefore bigger than the one before the crash. To remove those useless data, execute the REORGANIZE-ZIP-CONTAINER statement.

If several user IDs have access to the same container, the following constraint must be observered:

The backup file is always saved under the current user id. If the container is opened under a different user ID after the abnormal termination, the backup file cannot be found automatically. Therefore, the opening of the container open fails. In this case, the system administration has to locate the user ID of the backup file. The container then must be opened and restored under that user ID.

3 Command and statement interfaces

3.1 Starting and terminating the BS2ZIP applications

The BS2ZIP application has an SDF interface. If you enter a question mark, all possible inputs on the relevant level are displayed on the screen.
SDF is described in the manual „SDF Dialog Interface“. You will find the SDF syntax description in the “Commands” manual [2].

3.1.1 Command for starting the BS2ZIP application

The BS2ZIP application is started by means of the /START-ZIP-MANAGER command:

START-ZIP-MANAGER Alias: **ZIP-MANAGER**
VERSION = ***STD** / <product-version> ,**MONJV** = ***NONE** / <filename 1..54 without-gen-vers> ,**CPU-LIMIT** = ***JOB-REST** / <integer 1..32767 *seconds*>

VERSION = *STD / <product-version>
Specifies the BS2ZIP version which is to be used.
*STD is the default value: the version selected using the SELECT-PRODUCT-VERSION command is loaded. If no version was selected, the highest available version is loaded.

MONJV =
Name of the job variable that is to monitor the program.

MONJV = *NONE
No monitoring job variable is used.

MONJV = <filename 1..54 without-gen-vers>
Name of the job variable that is to monitor the program (see also „program monitoring“ in the “Job Variables” manual [3]).

CPU-LIMIT = *JOB-REST / <integer 1..32767>
Maximum CPU time, in seconds, allowed for execution of the program.
If the program execution exceeds the specified time, it is interrupted in interactive mode and message EXC0075 is issued. The user can then request a dump, abort the program or continue the program run. In batch mode the program is terminated.

*JOB-REST is the default value: no more than the maximum remaining CPU time is to be used. Detailed information about “Time limits in BS2000” is provided in the “Commands” manual [2].

3.1.2 Terminating the BS2ZIP application

Entering the END statement terminates the BS2ZIP application.

3.1.3 Messages

Information on program execution of BS2ZIP, processing the statements, and errors which have occurred is provided in messages of the message class SZP (message keys **SZP**xxxx).

The command or standard statement HELP-MSG-INFORMATION enables you to query more information on the meaning of the messages.

All messages of BS2ZIP can also be found using the HTML application “System messages” on the Manual Server (URL: http://b2manuals.ts.fujitsu.com).

3.2 Statements description

BS2ZIP application offers the following functions:

Statements	Meaning
ADD-FILE	Adds one or several BS2000 files to the currently opened ZIP archive.
CONVERT-ZIP-CONTAINER	Converts a ZIP container from PAM to SAM format with REC-FORM=U and vice versa (for data transfer using openFT < V11.0).
DELETE-FILE	Deletes one or several BS2000 files from the currently opened ZIP archive.
END	Closes the currently opened ZIP archive and stops the program.
EXTRACT-FILE	Extracts one or several files from the currently opened ZIP archive.
MODIFY-ZIP-OPTIONS	Defines defaults for BS2ZIP processing, e.g. for encryption
OPEN-ZIP-CONTAINER	Opens a ZIP archive. It eventually creates it if it does not exist.
REORGANIZE-ZIP-CONTAINER	Reorganizes the content of a ZIP container.
SHOW-FILE-ATTRIBUTES	Displays the contents of the currently opened ZIP archive.
START-TRACE	Activates an internal trace of the processing in a file.
STOP-TRACE	Terminates the trace processing.

In addition to these statements, you can also use SDF standard statements. These are displayed in a statement list which you are shown after you enter a question mark. The SDF standard statements are described in detail in the “SDF Dialog Interface” [1].

3.2.1 ADD-FILE

This statement adds one or several files to the currently opened ZIP file. The statement is rejected when the ZIP container is opened in read mode.

SAM, ISAM, PAM and PLAM files are supported. Elements of a PLAM library can also optionally be added to the ZIP container.

Prior to the version V21.0B10 BS2ZIP could only store original file metadata inside of a comment associated with a particular file in archive. This file comment starts with ***BS2:** string, which is followed by a comma-separated list of some of the DMS properties of the original file and is referred to as "file-info" in this manual. From V21.0B10 BS2ZIP also stores some metadata associated with the file in archive inside of an extensible data field. This new metadata includes:

- CCSNAME to which the original file was converted to by BS2ZIP during ADD-FILE.
- ISO code of the CCSNAME to which original file was converted to by BS2ZIP during ADD-FILE if applicable.
- if CCSNAME to which original file was converted to by BS2ZIP during ADD-FILE is based on EBCDIC or ASCII and if is a variant of Unicode.
- Type of line delimiter used.
- if the attribute CODED-CHARACTER-SET of the original file was *NONE.archive
- if standard conversion was made by BS2ZIP during ADD-FILE.
- if the file was added with DATA-TYPE=*BINARY (since V21.0B30)
- if the file encoding was marked as unknown by BS2ZIP when it was added to the archive, and character conversion is prohibited (since V21.0B32)

ADD-FILE

```
FROM-FILE = *ALL / *FROM-FILE(...) / *FROM-LIBRARY-ELEMENT(...) / *LIBRARY-ELEMENT(...) /
                    list-poss(20): <filename 1..54 without-gen-vers with-wild (80)>

  *FROM-FILE(...)

      |  LIST-FILE-NAME = <filename 1..54 without-gen-vers>

  *FROM-LIBRARY-ELEMENT(...)

      |  LIBRARY = <filename 1..54 without-gen-vers>

      |  ,ELEMENT = <composed-name 1..64 with-under>

  *LIBRARY-ELEMENT(...)

      |  LIBRARY = <filename 1..54 without-vers>

      |  ,ELEMENT = <composed-name 1..64 with-under with-wild(132)>(...)

      |       <composed-name 1..64 with-under with-wild(132)>(...)

      |           |  VERSION = *HIGHEST-EXISTING / *UPPER-LIMIT /

      |           |              <composed-name 1..24 with-under with-wild(40)>

      |           |  ,BASE = *STD / <composed-name 1..24 with-under with-wild(40)>
```

```
        |    ,TYPE = <alphanum-name 1..8 with-wild(12)>
,TO-FILE = *BY-SOURCE / *PATH-NAME(...) / <c-string 1..132 with-low> /
                <composed-name 1..98 with-underscore with-wild-constr(132)>(...)
    *PATH-NAME(...)
        |    PATH=<c-string 1..1800 with-low>
    <composed-name 1..98 with-underscore with-wild-constr(132)>(...)
        |    WITH-VERSION = *STD / *YES / *NO
        |    ,WITH-TYPE = *STD / *YES / *NO
,CHARACTER-CONVERSION = *BY-CONTAINER-FORMAT / *NO / *TO-EBCDIC / *TO-WIN-ANSI
,COMPRESSION-LEVEL = *STD / *NONE / *BEST-SPEED / *BEST-COMPRESSION
,DATA-TYPE = *NOT-SPECIFIED / *CHARACTER / *BINARY
,EXCEPT-FILE-NAME = *NONE / *FROM-FILE(...) / *FROM-LIBRARY-ELEMENT(...) /
                                list-poss(20): <filename 1..54 without-vers with-wild (80)>
    *FROM-FILE(...)
        |    LIST-FILE-NAME = <filename 1..54 without-gen-vers>
    *FROM-LIBRARY-ELEMENT(...)
        |    LIBRARY = <filename 1..54 without-gen-vers>
        |    ELEMENT = <composed-name 1..64 with-under>
,DELETE-SOURCE = *NO / *YES
,DELIMITER = *STD / *CRLF / *LF / *NL
,LOGGING = *MINIMUM / *MAXIMUM
```

FROM-FILE =
Specifies which files or which library elements are to be added to the ZIP container. The user can select individual files from the set of files specified here by means of the operand EXCEPT-FILE-NAME.

FROM-FILE = *ALL
All the supported files of the current userid are added to the ZIP file.

FROM-FILE = *FROM-FILE(...)
The path names of the files to be added to the ZIP container are to be taken from a file. The nonprivileged caller must be owner or co-owner of this file. This list file must be a SAM file with variable-length records containing one path name per record. Lower-case characters may be used, but they will be assimilated to upper-case characters. File names must be fully qualified with or without catid or userid.
The list file can be created, for instance, by means of the DMS command SHOW-FILE-ATTRIBUTES.

LIST-FILE-NAME = <filename 1..54 without-gen-vers>
Path name of the list file.

FROM-FILE = *FROM-LIBRARY-ELEMENT(...)
The path names of the files which are to be added to the ZIP container are to be taken from a PLAM library element (type S). The library element consists of records of variable length and contains one path name per record. Lower-case characters may be used, but they will be assimilated to upper-case characters. File names must be fully qualified with or without catid or userid.

LIBRARY = <filename 1..54 without-gen-vers>
Name of the PLAM library.

ELEMENT = <composed-name 1..64 with-under>
Name of the type-S element containing the list file. The element of the highest existing version is used.

FROM-FILE = *LIBRARY-ELEMENT(...)
Elements of a PLAM library are to be added to the ZIP container. An element is fully defined by its name, its type, and the version .

LIBRARY = <filename 1..54 without-vers>
Name of the PLAM library from which elements to be added.

ELEMENT = <composed-name 1..64 with-under with-wild(132)>(...)
The specified element is added to the ZIP container if the element type is supported. Wildcards enable more than one element to be specified.

VERSION = *HIGHEST-EXISTING / *UPPER-LIMIT / <composed-name 1..24 with-under with-wild(40)>
Version of the element which is to be displayed. The default value is *HIGHEST-EXISTING, i.e. the last element in alphabetical order.
*UPPER-LIMIT specifies the highest possible version X'FF' (displayed with the character '@').
If the version is specified using wildcards and library elements with the same name exist in versions which are affected by specifying wildcards, all these library elements are output.

BASE = *STD / <composed-name 1..24 with-under with-wild(40)>
Specifies the base if multiple data elements exist for the specified version.

TYPE = <alphanum-name 1..8 with-wild (12)>
Type of library element. The types D, J, M, P, S and X are supported, as well as the user-defined types based on these.
When the type is specified using wildcards, the name consists of up to 12 alphanumeric characters.

FROM-FILE = list-poss(20): <filename 1..54 without-vers with-wild(80)>
The path names of the files to be added to the ZIP container are specified directly. A list of up to 20 names may be specified.

The file names may be specified as fully or partially qualified names, with or without a catalog/user ID. If required, the file name is extended by the user ID of the request and the default catalog ID.

You can also use wildcard syntax to select the files. In this case, the supported files matching the pattern are added to the ZIP file.
If the specified file is temporary, then a shortened form can be used with '#'

TO-FILE = *BY-SOURCE
Specifies that the input files are to be registered in the container with their current name without catid or userid.

TO-FILE = *PATH-NAME(...)

The file is stored inside of the ZIP container with the specified name for this file. The operand will not interpret characters like forward slash, asterisk, square brackets and others as wildcards, but as part of the filename. Use this operand to specify a relative path name inside of zip container.

If FROM-FILE operand of ADD-FILE command uses wildcards, the input will be rejected if they return more than one item. The command can process multiple files at once if FROM-FILE operand is used with sub operands that imply that TO-FILE value is a prefix to destination filename in zip container. These sub operands of FROM-FILE include: *FROM-FILE, *FROM-LIBRARY-ELEMENT, list-poss(20).

PATH=<c-string 1..1800 with-low>

The path name of the file to be added to the ZIP container.

Example:

ADD-FILE FROM-FILE=*FROM-FILE(MYFILE),TO-FILE=*PATH-NAME('folder/prefix') is accepted.
All the file names contained in MYFILE are prefixed with folder/prefix inside of ZIP container.

TO-FILE = <c-string 1..132 with-low>

The file is stored inside of the ZIP container with the specified name for this file. Both lower and upper case characters of the operand will be accepted, because the operand is case sensitive. This operand can be used if the container is planned to be transferred to a non-BS2000 system and the file names concerned do not comply with BS2000 syntax (e.g. upper/lower case).

If wildcards are used in the FROM-FILE operand, wildcards can be used in a construction string to specify how the new names are to be formed in the container (see SDF rules for wildcard construction in the “Commands” manual [2]).

If a list of file names is specified in the FROM-FILE operand, only TO-FILE = *BY-SOURCE is accepted or a value without wildcards, but terminated by a period character (partial qualifier). In this case, the TO-FILE value is used as a prefix.

Example:

ADD-FILE FROM-FILE=*FROM-FILE(MYFILE),TO-FILE='XXX.' is accepted.
All the file names contained in MYFILE are prefixed with XXX.

TO-FILE = <composed-name 1..64 with-underscore with-wild-constr(132)>(...)

The file is stored inside of the ZIP container with the specified name for this file. The lower case characters of the operand will be accepted and changed to upper case, because the operand is not case sensitive.
If wildcards are used in the FROM-FILE operand, wildcards can be used in a construction string to specify how the new names are to be formed in the container (see SDF rules for wildcard construction in the “Commands” manual [2]).

If list of file names is specified in the FROM-FILE operand, only TO-FILE = *BY-SOURCE is accepted or a value without wildcards, but eventually terminated by a period character. In this case, the TO-FILE value is used as a prefix.

Example:

ADD-FILE FROM-FILE=*FROM-FILE(MYFILE),TO-FILE=XXX. is accepted.
All the file names contained in MYFILE are prefixed with XXX.

The default setting for library elements is that the specified name is also assigned a suffix containing the type and version of the element (format: <to-file>.<type>.<version>). The option of creating this suffix can be controlled in the following operands, which are available in TO-FILE only with **<composed-name 1..64 with-underscore with-wild-constr(132)>**:

WITH-VERSION = *STD / *YES / *NO
Evaluated only for library elements:
Specifies whether the suffix is to contain the element version.
*STD specifies *YES as the default.

An element with VERSION=*UPPER-LIMIT is assigned a 'U' in the suffix instead of the character '@' (e. g. in the case of type S, TO-FILE=myelem1 becomes MYELEM1.U.S).

WITH-TYPE = *STD / *YES / *NO
Evaluated only for library elements:
Specifies whether the suffix is to contain the element type.
*STD specifies *YES as the default.

CHARACTER-CONVERSION = *BY-CONTAINER-FORMAT
The input files are converted according to the format of the ZIP file.

Files are converted only

- if the archive is opened in winzip compatible format
- if the original file has SAM/ISAM structure
- if the DATA-TYPE is specified to *NOT-SPECIFIED or *CHARACTER.

Standard character conversion is applied only if the CODED-CHARACTER-SET of the original file is set to a CCSNAME of the *EBCDIC type; character conversion will not be performed on files with *ASCII or *UNICODE types. At first, BS2ZIP checks if the encoding type of the file is of type *EBCDIC by comparing its CODED-CHARACTER-SET to the values listed in the table below in the column *EBCDIC. If the encoding type is still unknown, the type is determined by XHCS. If the CCSNAME is undefined in XHCS, then it is considered of the *UNKNOWN type.

If CCSNAME is of type *UNKNOWN, only CHARACTER-CONVERSION = *NO can be applied; otherwise, the ADD-FILE statement ends with an error.

*EBCDIC	*ASCII	*UNICODE	*UNKNOWN

EDF03IRV EDF03DRV EDF04DRV EDF041 EDF042 EDF043 EDF044 EDF045 EDF046 EDF047 EDF049 EDF04A EDF04B EDF04C EDF04D EDF04E EDF04F EEHCL2 EEHCLC EEHCLC1 EEHCLAA EEHCLG	ISO88591 ISO88592 ISO88593 ISO88594 ISO88595 ISO88597 ISO88599 ISO8859F WCP1252P	UTF8 UTFE UTF16	All CCSNAMEs that are not in this table and that XHCS could not identify

Files inside winzip compatible archives with DATA-TYPE specified to *NOT-SPECIFIED or *CHARACTER and file structure of SAM or ISAM will have a line delimiter based on the encoding to which the original file was converted to or the original file encoding, if no conversion was applied, as described in the table below.

Encoding of file in archive	**Line delimiter with** **DELIMITER = *STD / *CRLF**	**Line delimiter with** **DELIMITER = *LF**	**Line delimiter with** **DELIMITER = *NL**
EBCDIC code (e.g. EDF041, EDF04F, UTFE)	0D25	25	15
ASCII code, except UTF16 (e.g. ISO88591, ISO8859F, WCP1252, UTF8)	0D0A	0A	0A

UTF16	000D000A	000A	000A

CHARACTER-CONVERSION = *NO
The input files are not converted.

Files inside winzip compatible archives with DATA-TYPE specified to *NOT-SPECIFIED or *CHARACTER and file structure of SAM or ISAM will have a line delimiter based on the encoding of the original file as described in the table below.

Files with unknown encoding type will be marked with a special flag inside the extensible data field. This flag prohibits character conversion during the extraction of such files.

Encoding of file in archive	**Line delimiter with DELIMITER = *STD / *CRLF**	**Line delimiter with DELIMITER = *LF**	**Line delimiter with DELIMITER = *NL**
EBCDIC code (e.g. EDF041, EDF04F, UTFE)	0D25	25	15
ASCII code, except UTF16 (e.g. ISO88591, ISO8859F, WCP1252, UTF8), unknown type (*UNKNOWN)	0D0A	0A	0A
UTF16	000D000A	000A	000A

CHARACTER-CONVERSION = *TO-EBCDIC
The ASCII encoded input files are converted to EBCDIC before compression.

Files are converted only

- if the archive is opened in winzip compatible format
- if the original file has SAM/ISAM structure are converted

Standard character conversion is applied only if the CODED-CHARACTER-SET of the original file is set to a CCSNAME of the *ASCII type; character conversion will not be performed on files with *EBCDIC or Unicode types. At first, BS2ZIP checks if the encoding type of the file is *ASCII by comparing its CODED-CHARACTER-SET to the values listed in the table above (in the CHARACTER-CONVERSION = *BY-CONTAINER-FORMAT section) in the column *ASCII. If the encoding type is still unknown, the type is determined by XHCS. If the CCSNAME is undefined in XHCS, then it is considered as a unknown type.

If CCSNAME has type *UNKNOWN, then only CHARACTER-CONVERSION = *NO can be applied; otherwise, ADD-FILE statement ends with an error.

Files inside winzip compatible archives with DATA-TYPE specified to *NOT-SPECIFIED or *CHARACTER and file structure of SAM or ISAM will have a line delimiter based on the encoding to which the original file was converted to or the original file encoding, if no conversion was applied, as described in the table below.

Encoding of file in archive	Line delimiter with DELIMITER = *STD / *CRLF	Line delimiter with DELIMITER = *LF	Line delimiter with DELIMITER = *NL
EBCDIC code (e.g. EDF041, EDF04F, UTFE)	0D25	25	15
ASCII code, except UTF16 (e.g. ISO88591, ISO8859F, WCP1252, UTF8)	0D0A	0A	0A
UTF16	000D000A	000A	000A

CHARACTER-CONVERSION = *TO-WIN-ANSI
The EBCDIC encoded input files are converted to WIN-ANSI before compression.

The logic of CHARACTER-CONVERSION =*TO-WIN-ANSI is identical to CHARACTER-CONVERSION =*BY-CONTAINER-FORMAT, except if DATA-TYPE = *BINARY conversion will be performed.

Files are converted only

- if the archive is opened in winzip compatible format
- if the original file has SAM/ISAM structure are converted

Standard character conversion is applied only if the CODED-CHARACTER-SET of the original file is set to a CCSNAME of the *EBCDIC type; character conversion will not be performed on files with *ASCII or Unicode types. At first, BS2ZIP checks if the encoding type of the file is *EBCDIC by comparing its CODED-CHARACTER-SET to the values listed in the table above (in the CHARACTER-CONVERSION = *BY-CONTAINER-FORMAT section) in the column *EBCDIC. If the encoding type is still unknown, the type is determined by XHCS. If the CCSNAME is undefined in XHCS, then it is considered as a unknown type.

If CCSNAME has type *UNKNOWN, then only CHARACTER-CONVERSION = *NO can be applied; otherwise, ADD-FILE statement ends with an error.

Files inside winzip compatible archives with DATA-TYPE specified to *NOT-SPECIFIED or *CHARACTER and file structure of SAM or ISAM will have a line delimiter based on the encoding to which the original file was converted to or the original file encoding, if no conversion was applied, as described in the table below.

Encoding of file in archive	Line delimiter with DELIMITER = *STD / *CRLF	Line delimiter with DELIMITER = *LF	Line delimiter with DELIMITER = *NL
EBCDIC code (e.g. EDF041, EDF04F, UTFE)	0D25	25	15
ASCII code, except UTF16 (e.g. ISO88591, ISO8859F, WCP1252, UTF8)	0D0A	0A	0A
UTF16	000D000A	000A	000A

COMPRESSION-LEVEL =
This operand determines the compression level. It allows adjusting the speed/compression size ratio. This operand is inapplicable to empty files as they are stored without compression as they only require to store their metadata.

COMPRESSION-LEVEL = *STD
This compression mode selects a compromise between the speed and the compression size.

COMPRESSION-LEVEL = *BEST-SPEED
This compression mode selects the fastest compression. Such a mode results in a larger container.

COMPRESSION-LEVEL = *BEST-COMPRESSION
This compression mode selects the best compression. Such a mode results in a smaller container but requires larger processing time.

COMPRESSION-LEVEL = *NONE
The file is added without compression.

DATA-TYPE =
The DATA-TYPE operand controls the usage and line delimiter type for text files. This option is only relevant for SAM files to be added to a winzip compatible container. The default value *NOT-SPECIFIED currently corresponds to *CHARACTER.

DATA-TYPE	Container format WINZIP compatible	Container format BS2000
*NOT-SPECIFIED / *CHARACTER	• SAM/ISAM files: assumed as text files. A record = a line of text. Line delimiter depends on CCSNAME. • PAM files: Assumed as binary files.	• SAM/ISAM files: assumed as text files. A record = a line of text. Line delimiter is universal and does not depend on CCSNAME. • PAM files: Assumed as binary files.
*BINARY	• SAM files: Assumed as binary files. The file can only be extracted as PAM file. • ISAM files: Not relevant. Assumed as text files. A record = A line of text • PAM files: Processed as binary files.	Not relevant. Processed as DATA-TYPE=*NOT-SPECIFIED

EXCEPT-FILE-NAME =
Serves to specify files that are to be excluded from zipping.

EXCEPT-FILE-NAME = *NONE
All files specified with the FROM-FILE operand are to be archived.

EXCEPT-FILE-NAME = *FROM-FILE(...)
The path names of the files to be excluded from zipping are to be taken from a file. The nonprivileged caller must be owner or co-owner of this file. This list file must be a SAM file with variable-length records containing one path name per record. Lower-case characters may be used, but they will be assimilated to upper-case characters. File names must be fully qualified with or without catid or userid.
The list file can be created, for instance, by means of the DMS command SHOW-FILE-ATTRIBUTES.

LIST-FILE-NAME = <filename 1..54 without-gen-vers>
Path name of the list file.

EXCEPT-FILE-NAME = *FROM-LIBRARY-ELEMENT(...)
The path names of the files which are not to be zipped are taken from a PLAM library element (type S). The library element consists of records of variable length and contains one path name per record. Lower-case characters may be used, but they will be assimilated to upper-case characters. File names must be fully qualified with or without catid or userid.

LIBRARY = <filename 1..54 without-gen-vers>
Name of the PLAM library.

ELEMENT = <composed-name 1..64 with-under>
Name of the type-S element containing the list file. The element of the highest existing version is used.

EXCEPT-FILE-NAME = list-poss(20): <filename 1..54 without-vers with-wild(80)>
The path names of the files to be excluded from archival are specified directly. A list of up to 20 names may be specified.

The first character of the file names must not be a hyphen. The file names may be specified as fully or partially qualified names, with or without a catalog/user ID. If required, the file name is extended by the user ID of the request and the default catalog ID.

You can also use wildcard syntax to select the files.

DELETE-SOURCE = *NO / *YES
Specifies whether the original files/library elements are to be deleted after they have been added to the ZIP archive. The default *NO retains the original files/library elements .

DELETE-SOURCE = *YES
The original files/library elements are deleted. A message is issued if deletion is not possible because of the protection attributes.

i
- If the source file is protected against writing, the file is added in the ZIP container and an error message is sent to the user.
- If the source library element is protected against writing, the library element is added in the zip container and an error message is sent to the user.
- If the source library is protected against writing, the library element is added in the zip container and an error message is sent to the user.

DELIMITER = *STD / *C**RLF / *L**F / *N**L**

Controls the line delimiter for text files added to winzip compatible archive. This operand is ignored if:

- The archive is in BS2000 format
- DATA-TYPE=*BINARY
- FCBTYPE of the original file is PAM

The operand DELIMITER will pick the line delimiter based on CCSNAME of the encoding in which the file is stored inside of the archive.

DELIMITER operand value	ASCII (except UTF16)	EBCDIC	UTF16
*STD or *CRLF	0D0A	0D25	000D000A
*LF	0A	25	000A
*NL	0A	15	000A

LOGGING = *MINIMUM / *MAXIMUM
Controls the amount of the message output.

LOGGING = *MINIMUM
Only error messages will be sent.

LOGGING =*MAXIMUM
All messages will be sent. Currently the [guaranteed] messages SZP0116 and SZP0117 are sent after resp. file addition and library element addition; further messages may be added in the future.

Notes

- Compression figures:

Compression mode	Size after compression (280630 PP uncompressed)	Compression ratio	CPU time
*BEST-SPEED	54096 PP	80,7 %	448,5 sec.
*STD	44496 PP	84,2 %	970,2 sec.
*BEST-COMPRESSION	43488 PP	84,5 %	8229,6 sec.

- If the K2 key is pressed during the ADD-FILE statement, processing is interrupted with the query message `SZP0208`:
  - The user can simply continue processing.
  - The user can terminate processing and return to statement mode (//). The files which had not been added by the time the interruption occurred are not added. If required, they must be added again.

- Using the default TO-FILE=*BY-SOURCE file names are always registered in the ZIP container without catid /userid. To register catid/userid, they have to be specified in the TO-FILE operand.

Specification in ADD-FILE	**Registered as**
FROM-FILE = MYFILE	MYFILE
FROM-FILE = $UID.MYFILE	MYFILE
FROM-FILE = :CAT1:$UID.MYFILE	MYFILE
FROM-FILE = MYFILE, TO-FILE=$UID.MYFILE	$UID.MYFILE
FROM-FILE = MY*,TO-FILE=NEW-*	NEW-FILE
FROM-FILE = #MYFILE.1	S.163.TSN1.MYFILE.1
FROM-FILE = #MYFILE.1,TO-FILE=MYFILE	MYFILE
FROM-FILE = #MYFILE.*,TO-FILE=MYFILE-*	MYFILE-1

- If data encryption had been set using the MODIFY-ZIP-OPTIONS statement, the files are encrypted when they are added. The standard Zip 2.0 encryption used here is compatible with WinZip on Windows-based systems. Encryption is inapplicable to empty file as they are stored without encryption as they only require to store their metadata.

3.2.2 CONVERT-ZIP-CONTAINER

By default a ZIP container has PAM file format.

Alternatively a ZIP container can also be a binary SAM file (with RECORD-FORMAT=U). This format occurs in the following cases (see also section “Windows interoperability”):

- Only if the ZIP container was transferred to BS2000 using openFT < V11.0 will it have been stored as a binary SAM file (i.e. with RECORD-FORMAT=U). For processing purposes the ZIP container must be in PAM file format.

The CONVERT-ZIP-CONTAINER statement converts a ZIP container to the other file format (from PAM to SAM with RECORD-FORMAT=U and vice versa). The ZIP container is retained unchanged, and the converted ZIP container is stored in the output file specified.

CONVERT-ZIP-CONTAINER

```
FROM-FILE = *CURRENT / <filename 1..54 without-gen-vers> / *LINK(...)
      *LINK(...)
            |     LINK-NAME = <structured-name 1..8>
,TO-FILE = *STD / <filename 1..54 without-gen-vers> / *LINK(...)
      *LINK(...)
            |     LINK-NAME = <structured-name 1..8>
,WRITE-MODE = *CREATE / *REPLACE-ONLY / *ANY
```

FROM-FILE = *CURRENT / <filename 1..54 without-gen-vers> / *LINK(...)
Specifies the ZIP container which is to be converted.

FROM-FILE = *CURRENT
The container is the one which is presently open.

FROM-FILE = <filename 1..54 without-gen-vers>
Converts the specified ZIP container.

FROM-FILE = *LINK(...)
The ZIP container to be converted is specified by means of a link name.

LINK-NAME = <structured-name 1..8>
Link name which is assigned to the ZIP container.

TO-FILE = *STD / <filename 1..54 without-gen-vers> / *LINK(...)
Specifies the output file in which the converted ZIP file is stored.

TO-FILE = *STD
The output file is assigned a standard name which contains the file name of the ZIP container which is to be converted and a suffix indicating the file format involved (SAM or PAM).
The output file is created under the user-id and default cat-id of the user. An user-id and/or cat-id specified in the FROM-FILE parameter will be ignored for the output file.

The standard name is formed in accordance with the following rules:

- The ZIP container is in PAM file format.

Name of the ZIP container	Name of the output file
partialname.PAM	partialname.SAM
partialname.PAM.ZIP	partialname.SAM.ZIP
partialname.ZIP	partialname.SAM.ZIP
filename	filename.SAM

- The ZIP container is in SAM file format with RECORD-FORMAT=U.

Name of the ZIP container	Name of the output file
partialname.SAM	partialname.PAM
partialname.SAM.ZIP	partialname.PAM.ZIP
partialname.ZIP	partialname.PAM.ZIP
filename	filename.PAM

In the case of a *partialname*.ZIP or *filename* ZIP container, the path name for the output file can be longer than 54 characters when the suffix is appended. In this case the *partialname* or *filename* in front of the suffix is truncated by the required number of characters.

TO-FILE = <filename 1..54 without-gen-vers>
The converted ZIP container is stored in the specified file.

TO-FILE = *LINK(...)
The output file is specified via a link name.

LINK-NAME = <structured-name 1..8>
Link name which is assigned to the output file.

WRITE-MODE =
Specifies whether the output file should be created or only overwritten.

WRITE-MODE = *CREATE
The output container must be created. The statement is rejected if it already exists.

WRITE-MODE = *REPLACE-ONLY
The output file must exist and is replaced. The statement is rejected if the file does not exist.

WRITE-MODE = *ANY

The output file is overwritten if it exists, otherwise it is created.

3.2.3 DELETE-FILE

This statement deletes one or several files from the ZIP file which is currently open. The statement is rejected when the ZIP container is opened in read mode.

The size of the ZIP container does not change when files contained in it are deleted. Memory space which is no longer required is released only when the ZIP container is reorganized (see the OPEN-ZIP-CONTAINER statement).

DELETE-FILE

FILE-**NAME** = ***ALL** / <composed-name 1..98 with-under with-wild(132)> / <c-string 1..1024 with-low> / ***P**ATH-**N**AME(...)

***P**ATH-**N**AME(...)

| PATH=<c-string 1..1800 with-low>

FILE-NAME = *ALL
All the files are deleted from the ZIP file.

FILE-NAME = <composed-name 1..98 with-under with-wild(132)>
The specified file is deleted from the ZIP file. When wildcards are used, all files matching the pattern are deleted from the ZIP file.

FILE-NAME = <c-string 1..1024 with-low>
All files which match the specified string (wildcards according to the SDF rules for wildcard selection (see SDF syntax in the “Commands” manual [2]) are permitted) are deleted from the ZIP container. Specification as a C string must be used if the container was created in a non-BS2000 system and the file names concerned do not comply with BS2000 syntax (e.g. upper/lower case).

FILE-NAME = *PATH-NAME(...)

The specified file is deleted from the ZIP file. The operand will not interpret characters like forward slash, asterisk, square brackets and others as wildcards, but as part of the filename. Use this operand to specify a relative path name inside of zip container.

3.2.4 END

This statement closes the currently opened ZIP file and terminates the program.

```
END
```

3.2.5 EXTRACT-FILE

This statement extracts files from the currently opened ZIP file.

EXTRACT-FILE

FILE-NAME = ***ALL** / <composed-name 1..98 with-under with-wild(132)> / <c-string 1..1024 with-low> / ***P**ATH-**N**AME(...)

***P**ATH-**N**AME(...)

| PATH=<c-string 1..1800 with-low>

, **TO-F**ILE = ***BY-SOURCE** / <filename 1..54 without-gen-vers with-wild-constr(80)>

, **WR**ITE-**MODE** = ***CREATE** / ***REPLACE-ONLY** / ***ANY**

, **DATA-TYPE** = ***NOT-SPECIFIED** / ***CHAR**ACTER / ***BIN**ARY / ***SAM-BIN**ARY

, **CHARACTER-CONVERSION** = ***BY-CONTAINER-FORMAT** / ***NO** / ***TO-WIN-ANSI** / ***TO-EBCDIC** / ***BY-PARAMETERS(...)**

***BY-PARAMETERS(...)**

| **FROM-CCS** = **WCP1252P /** < name 1..8>

| **,TO-CCS** = **EDF04F /** < name 1..8>

, **BLOCK-CONTR**OL-**INFO** = ***KEEP** / ***IGNORE**

,**P**AD-**E**MPTY-**R**ECORD = ***NO** / ***Y**ES

,DELIMITER = ***S**TD / ***C**RLF / ***L**F / ***N**L / ***0D0**A / ***0A / *0D2**5 / ***2**5 / ***1**5 / ***000D**000A / ***000A**

,LOGGING = ***MINIMUM** / ***MAXIMUM**

FILE-NAME = *ALL
All the files included in the container are extracted.

FILE-NAME = <composed-name 1..98 with-under with-wild(132)>
The specified file is extracted from the container. When wildcards are used, all files matching the pattern are extracted from the container.

FILE-NAME = <c-string 1..1024 with-low>
All files which match the specified string (wildcards according to the SDF rules for wildcard selection (see SDF syntax in the “Commands” manual [2]) are permitted) are extracted from the ZIP container. Specification as a C string must be used if the container was created in a non-BS2000 system and the file names concerned do not comply with BS2000 syntax (e.g. upper/lower case).

FILE-NAME = *PATH-NAME(...)

The specified file is extracted from the container. The operand will not interpret characters like forward slash, asterisk, square brackets and others as wildcards, but as part of the filename. Use this operand to specify a relative path name inside of the ZIP container.

TO-FILE = *BY-SOURCE / <filename 1..54 without-gen-vers with-wild-constr(80)>

According to the origin of the zipped files, the output name will respect the following rules:

- For BS2000 files, the output name is built according to the SDF rules for wildcard construction (see SDF syntax in the “Commands” manual [2]).
- For other files (PC, Unix system), the output name is built by replacing the '*' character in the TO-FILE operand by the file name registered in the container. The eventual path of the file is ignored in the file name construction process.

TO-FILE=	**File origin BS2000**	**File origin not BS2000**
*BY-SOURCE	The output file name will be the file name as registered in the container. • If the file has been registered with a catid/userid, it will be extracted under this catid /userid. • If the file has been registered without catid/userid, it is extracted under the current userid/catid.	The output filename will be the file name without access path prefixed by the current catid/userid. File names which do not comply with the syntax in BS2000 are renamed by BS2ZIP (see below).
<filename 1..54 without-gen-vers with-wildconstr(80)>	Valid format is: • a file name without wildcard. • a file name with wildcards: the output name is built according to the SDF rules for wildcard construction. If the new file name does not respect BS2000 syntax, the extract of the file is rejected.	Valid format is: • a file name without wildcard. • [<PREFIX>]*[<SUFFIX>]: the output file name will be [<PREFIX>]filename[<SUFFIX>] where file name is the registered file name without directory path. If the resultant file name does not comply with the syntax in BS2000, it is renamed by BS2ZIP (see below).

If some zipped file names are not BS2000 compliant, it is necessary to extract them one by one, specifying for each of them a valid output name. If extraction results in an invalid file name, the file concerned is assigned the following alternative file name:

FILExxxx.yyyymmdd.hhmmss

where xxxx is a sequence number, yyyymmdd.hhmmss is the current date and time. BS2ZIP shows that the file has been renamed with the message `SZP0090`. Example:

```
% SZP0090 Warning. File name 'TEST_KDO_1.HTM' is not BS2000 compliant.
The file will be extracted under the name 'FILE0001.20111108.161442'
```

The catid /userid of those files are taken from the <PREFIX>. If they are not specified the catid/userid of the current user are used by default.

WRITE-MODE =

Controls whether an existing file with the same name as output file can be replaced. CODED-CHARACTER-SET of the existing file will infer current encoding and encoding after character conversion of the extracted file from an archive. Depending on CHARACTER-CONVERSION operand the following conversion will be performed:

CHARACTER-CONVERSION operand	C-C-S of existing file	Current encoding of file in archive	Character conversion type
*BY-CONTAINER-FORMAT	Non-UTF EBCDIC (EDF041, EDF042, ...)	Non-UTF ASCII (ISO88591, ISO88592, ...)	ASCII -> EBCDIC
*BY-CONTAINER-FORMAT	Non-UTF ASCII (ISO88591, ISO88592, ...)	Same non-UTF ASCII (ISO88591, ISO88592, ...)	No conversion
*BY-CONTAINER-FORMAT	UTF (UTF8, UTF16, UTFE)	Same encoding	No conversion
*NO	Any encoding	Same encoding	No conversion
*TO-WIN-ANSI	Non-UTF ASCII (ISO88591, ISO88592, ...)	Non-UTF EBCDIC(EDF041, EDF042, ...)	EBCDIC-> ASCII
*TO-WIN-ANSI	Non-UTF EBCDIC (EDF041, EDF042, ...)	Extraction error	Extraction error
*TO-WIN-ANSI	UTF (UTF8, UTF16, UTFE)	Same UTF (UTF8, UTF16, UTFE)	No conversion
*TO-EBCDIC	Non-UTF ASCII (ISO88591, ISO88592, ...)	Extraction error	Extraction error
*TO-EBCDIC	Non-UTF EBCDIC (EDF041, EDF042, ...)	Non-UTF ASCII (ISO88591, ISO88592, ...)	ASCII -> EBCDIC
*TO-EBCDIC	UTF (UTF8, UTF16, UTFE)	Same UTF (UTF8, UTF16, UTFE)	No conversion
*BY-PARAMETERS(FROM-CCS,TO-CSS)	Any encoding (existing file is ignored)	FROM-CCS value	FROM-CCS -> TO-CSS

WRITE-MODE = *CREATE

The output file must not exist and is created. The statement is rejected if the file exists already.

WRITE-MODE = *REPLACE-ONLY

The output file must exist and is replaced. The statement is rejected if the file does not exist. If BS2ZIP extracts a foreign text file it will check CODED-CHARACTER-SET attribute of the replaced existing file. This attribute will be used an encoding of the extracted file. BS2ZIP also will infer CCSNAME of the encoding in which the file is stored inside of the archive from this attribute as it is described in the table above.

WRITE-MODE = *ANY

The output file is overwritten if exists or created otherwise. If BS2ZIP extracts a foreign text file and replaces an existing file it will check CODED-CHARACTER-SET attribute of the replaced file. This attribute will be used as encoding of the extracted file. BS2ZIP also will infer CCSNAME of the encoding in which the file is stored inside of the archive from this attribute as it is described in the table above.

DATA-TYPE =

This operand controls the record structure of the files to be extracted.

DATA-TYPE	Container format WINZIP compatible	Container format BS2000
*NOT-SPECIFIED	If no file-info found in ZIP container then *CHARACTER is assumed. Else file-info is used and character conversion is done or not according to file type and CHARACTER-CONVERSION operand	If no file-info found Then error Else file-info is used - no conversion
*CHARACTER	File is extracted as a SAM file Records are assumed to be delimited by 0D0A Character conversion is performed according to the CHARACTER-CONVERSION operand	Statement rejected
*BINARY	File is extracted as a PAM file No conversion is performed. In case of original PAM files, the extract is rejected with error message SZP0121	Statement rejected
*SAM-BINARY	File is extracted as a binary SAM file (REC-FORM=U) No conversion performed	Statement rejected

CHARACTER-CONVERSION =

This operand controls the WIN-ANSI/EBCDIC conversion. It is only supported for the following files:

- files extracted with DATA-TYPE=*CHARACTER
- files extracted with DATA-TYPE=*NOT-SPECIFIED and which are not indicated as PAM file in the file-info.

Type of conversion is determined by the presence and values of the CODED-CHAR-SET attribute and of the extensible data field. This metadata as well as which encoding type it is interpreted as can be checked with the SHOW-FILE-ATTRIBUTES statement with the operand INFORMATION=*ALL.

CHARACTER-CONVERSION		Conversion behaviour
*BY-CONTAINER-FORMAT		Default value. Behavior determined by the ZIP container open format. Attempts to restore to the original file encoding.
*NO		No conversion is performed
*TO-WIN-ANSI		WIN-ANSI conversion is performed (only for extraction to SAM/ISAM file)
*TO-EBCDIC		EBCDIC conversion is performed (only for extraction to SAM/ISAM file)
*BY-PARAMETERS (...)		Conversion will be made according to specified source and target character sets (only for extraction to SAM/ISAM file)
	FROM-CCS	Origin coded character set, from which all characters will be converted.

	TO-CCS= *STD	Target coded character set is picked by finding the first fully compatible coded character set in EBCDIC of the same ISO code variant number as origin coded character set. If origin coded character set is Unicode, than no conversion is performed. (ISO code variant number, see manual XHCS [5]).
	TO-CCS	Target coded character set, to which all characters will be converted.

The type of conversion is determined by:

- CHARACTER-CONVERSION operand value
- DATA-TYPE operand value
- WRITE-MODE operand and CODED-CHARACTER-SET attribute of the replaced file
- the presence and the value of the FCBTYPE attribute inside of the file-info
- the presence and the value of the CODED-CHAR-SET attribute inside of the file-info
- the presence and the value of the extensible data field.
- Scan of the first 32 kB of the file (It is performed only if the CODED-CHAR-SET attribute inside of the file-info is not present)
- the presence of the special flag marking unknown encoding type

Metadata of a file in an archive can be checked with SHOW-FILE-ATTRIBUTES statement with operand INFORMATION=*ALL (COMMENTS for the file-info and CCSNAME for the extensible data field). File-info with file attributes should be contained in the metadata of all files in archive added by BS2ZIP. File-info contains an attribute CODED-CHAR-SET, which is the original encoding of added files. If the file-info is not present in the metadata of file in archive, then it is assumed that the file is a foreign ; i.e. it was added by third party archive manager from open systems. By default such files are scanned to check the most likely encoding between ISO8859F, WCP1252, UTF8, UTF16. The encoding detection algorithm is probabilistic and it is recommended to set the type character conversion manually. Starting from V21.B10 the extensible data field is added by BS2ZIP to metadata of all files in archives by BS2ZIP. It contains a field with the name of the encoding in which the file is stored inside of the archive.

Character conversion of files in Unicode variants is only available with CHARACTER-CONVERSION=*BY-PARAMETERS(). Extraction of Unicode files with all other values of this operand results in character conversion being skipped. Supported Unicode CCSNAMEs: UTF8, UTFE, UTF16 (Big-endian).

All CCSNAMEs in BS2ZIP are divided into 4 types: single byte EBCDIC (*EBCDIC), single byte ASCII (*ASCII), Unicode (*UNICODE) and unknown (*UNKNOWN). Below is the table with the list of encodings for each type:

*EBCDIC	*ASCII	*UNICODE	*UNKNOWN

EDF03IRV	ISO88591	UTF8	All CCSNAMEs that
EDF03DRV	ISO88592	UTFE	are not in this table
EDF04DRV	ISO88593	UTF16	and that XHCS could
EDF041	ISO88594		not identify
EDF042	ISO88595		
EDF043	ISO88597		
EDF044	ISO88599		
EDF045	ISO8859F		
EDF046	WCP1252P		
EDF047			
EDF049			
EDF04A			
EDF04B			
EDF04C			
EDF04D			
EDF04E			
EDF04F			
EEHCL2			
EEHCLC			
EEHCLC1			
EEHCLAA			
EEHCLG			

If a file was added by BS2ZIP to an archive with the unknown type, then EXTRACT-FILE can only be used with CHARACTER-CONVERSION = *NO to disable character conversion, or CHARACTER-CONVERSION = *BY-PARAMETERS to ignore the metadata of the file and specify the current encoding and desired encoding of the output file. All other options for CHARACTER-CONVERSION will result in an error for the EXTRACT-FILE statement.

CHARACTER-CONVERSION = *BY-CONTAINER-FORMAT
Default value. Behavior determined by the ZIP container open format. BS2ZIP attempts to restore to the original file encoding.

Files are converted only

- if the archive is opened in winzip compatible format
- if the original file has SAM/ISAM structure are converted
- if DATA-TYPE is specified to *NOT-SPECIFIED or *CHARACTER

- file is not in Unicode

Condition	Result
• Archive is opened in BS2000 format • Original file has PAM structure • DATA-TYPE is specified to *BINARY or *SAM-BINARY • File is in Unicode	No character conversion is performed.
File in archive has: • Extensible data field • File-info with CODED-CHAR-SET attribute	Character conversion from current encoding to the original file encoding is performed[1].
File in archive has: • File-info with CODED-CHAR-SET attribute • CODED-CHAR-SET attribute that starts with a string 'EDF' But does not have: • Extensible data field	Conversion from ISO8859F to EDF04F
File in archive has: • File-info with CODED-CHAR-SET attribute • CODED-CHAR-SET attribute that does not start with a string 'EDF' But does not have: • Extensible data field	No character conversion is performed.
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *CREATE or *ANY that does not overwrite existing file • Scan of the first 32 kB of file detected Unicode (UTF8 or UTF16)	No character conversion is performed.

• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *CREATE or *ANY that does not overwrite existing file • Scan of the first 32 kB of file detected non-Unicode (ISO8859F or WCP1252).	Character conversion to EBCDIC is performed[1] .
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file • CODED-CHARACTER-SET attribute of existing file is set to non-Unicode EBCDIC (e.g. EDF041, EDF042, ...)	Character conversion from ASCII to EBCDIC is performed[1].
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file • CODED-CHARACTER-SET attribute of existing file is set to Unicode or ASCII (e.g. UTF16, UTFE, ISO88591, ...)	No character conversion is performed.

1 see info box below

CHARACTER-CONVERSION = *NO

No character conversion is performed. Metadata of a file in an archive only impacts CODED-CHARACTER-SET attribute of the output file. If neither extensible data field, nor file-info with CODED-CHAR-SET attribute are present, then BS2ZIP performs a scan of the first 32 kiB of the data to determine the most likely encoding between ISO8859F, WCP1252, UTF8, UTF16.

Condition	Result
• File in archive has extensible data field	CODED-CHARACTER-SET attribute of the output file is set to CODED-CHAR-SET attribute of the file in the archive .
• If file in archive has file-info with CODED-CHAR-SET attribute	CODED-CHARACTER-SET attribute of the output file is set to CODED-CHAR-SET attribute of the file in the archive .

• If file in archive does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *CREATE or *ANY that does not overwrite existing file	CODED-CHARACTER-SET attribute of the output file is set based on scan of the first 32 kB of the file in the archive . Possible values: ISO8859F, WCP1252, UTF8, UTF16
• If file in archive does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file	CODED-CHARACTER-SET attribute of the output file remains preserved.

CHARACTER-CONVERSION = *TO-WIN-ANSI

WIN-ANSI conversion is performed. The output file will always be in ASCII, except when file in archive is stored in a Unicode variant.

Condition	Result
• Original file has PAM structure • DATA-TYPE is specified to *BINARY or *SAM-BINARY • File is in Unicode	No character conversion is performed.
File in archive has: • Extensible data field is present • File-info with CODED-CHAR-SET attribute	Character conversion from current encoding to ASCII is performed unless it is already in ASCII (according to extensible data field)*.
File in archive has: • File-info with CODED-CHAR-SET attribute • CODED-CHAR-SET attribute that starts with a string 'EDF' But does not have: • Extensible data field	Conversion from EDF04F to ISO8859F.

File in archive has: • File-info with CODED-CHAR-SET attribute • CODED-CHAR-SET attribute that does not start with a string 'EDF' But does not have: • Extensible data field	No character conversion is performed.
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *CREATE or *ANY that does not overwrite existing file	Conversion from EDF04F to ISO8859F.
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file • CODED-CHARACTER-SET attribute of existing file is set to non-Unicode EBCDIC (e.g. ISO88591, ISO88592, ...)	Character conversion from EBCDIC to ASCII is performed[1] .
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file • CODED-CHARACTER-SET attribute of existing file is set to Unicode or EBCDIC (e.g. UTF16, UTFE, EDF041, ...)	No character conversion is performed.

1 see info box below

CHARACTER-CONVERSION = *TO-EBCDIC

EBCDIC conversion is performed. The output file will always be in EBCDIC, except when file in archive is stored in Unicode variant.

Condition	Result

• Original file has PAM structure • DATA-TYPE is specified to *BINARY or *SAM-BINARY • File is in Unicode	No character conversion is performed.
File in archive has: • Extensible data field is present • File-info with CODED-CHAR-SET attribute	Character conversion from current encoding to EBCDIC is performed unless it is already in EBCDIC (according to extensible data field)*.
File in archive has: • File-info with CODED-CHAR-SET attribute • CODED-CHAR-SET attribute that starts with a string 'EDF' But does not have: • Extensible data field	Conversion from ISO8859F to EDF04F.
File in archive has: • File-info with CODED-CHAR-SET attribute • CODED-CHAR-SET attribute that does not start with a string 'EDF' But does not have: • Extensible data field	No character conversion is performed.
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *CREATE or *ANY that does not overwrite existing file	Conversion from ISO8859F to EDF04F.
• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file • CODED-CHARACTER-SET attribute of existing file is set to non-Unicode EBCDIC (e.g. EDF041, EDF042, ...)	Character conversion from ASCII to EBCDIC is performed[1].

• File does not have file-info with CODED-CHAR-SET attribute • WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file • CODED-CHARACTER-SET attribute of existing file is set to Unicode or ASCII (e.g. UTF16, UTFE, ISO88591, ...)	No character conversion is performed.

1 see Info box below.

CHARACTER-CONVERSION = *BY-PARAMETERS(...)

Conversion will be made according to specified source and target character sets.

Files are converted only

- if the original file has SAM/ISAM structure
- if DATA-TYPE is specified to *NOT-SPECIFIED or *CHARACTER.

This option allows to ignore metadata of a file in archive in order to set the character conversion type manually.

During extraction of an file from archive this option allows to ignore:

- CODED-CHAR-SET attribute inside of the file-info
- Extensible data field
- CODED-CHARACTER-SET attribute of existing file when WRITE-MODE = *REPLACE-ONLY or *ANY that overwrites existing file.

It is particularly useful for files added to archive by different archive managers in order to skip foreign file conversion scan of the first 32 kB. FROM-CCS and TO-CSS accept all CCSNAME values that are available, unless they both are not Unicode variant and have different ISO codes. Specifying the same value for both FROM-CSS and TO-CCS leads to BS2ZIP skipping conversion, but it will also specify this CCSNAME as CODED-CHARACTER-SET attribute of the extracted file.

TO-CSS also accepts *STD. If FROM-FILE is not Unicode and is ASCII, then BS2ZIP will pick a pair EBCDIC CCSNAME of the same ISO code and convert to it. In all other cases with TO-CSS=*STD conversion will be skipped, but BS2ZIP will also specify CCSNAME from FROM-CCS as CODED-CHARACTER-SET attribute of the extracted file.

If XHCS was not able to convert some of the characters, a warning message will be issued and these characters will be set to dot '.' (x'4B').

Metadata of file in archive can contain CCSNAMEs within that are not present on the system currently. Extensible data field contains information about ISO codes of the encoding, whether it was EBCDIC /ASCII, Unicode or not. This information will be used to substitute encodings to their best available alternative defined on the system.

For example: according to metadata of a file in an archive its current encoding is WCP1252 (ISO code 15), while the original file encoding was EDF04F, but WCP1252 is not currently defined in XHCS. BS2ZIP calls XHCS to check available CCSNAMEs with ISO code 15; it sees that ISO8859F is available and converts the file from ISO8859F to EDF04F during file extraction.

BLOCK-CONTROL-INFO =
Controls the block control attribute of the resulting file. This allows particularly to extract original PAMKEY file on a NK disk.

BLOCK-CONTROL-INFO = *KEEP
The resulting file keeps the same block control attribute than the original file

BLOCK-CONTROL-INFO = *IGNORE
The resulting file is created with the default block control of the disk where it is saved.

PAD-**E**MPTY-**R**ECORD = ***N**O / ***Y**ES
Controls if empty lines are padded with a blank character when extracted from a winzip compatible archive. Padding is only applied when DATA-TYPE is *NOT-SPECIFIED or *CHARACTER.

DELIMITER =
Controls which line delimiter is assumed to be separating the lines in the extracted file. This setting is ignored unless DATA-TYPE is *NOT-SPECIFIED or *CHARACTER and the archive is in winzip compatible format.

If this option is set to ***S**TD / ***C**RLF / ***L**F / ***N**L, then the delimiter option is based on the stored encoding of the file in archive and CHARACTER-CONVERSION option. Delimiters associated with different encodings are in the following table:

Delimiter	Single byte ASCII, UTF8	EBCDIC	UTF16
***C**RLF	0D0A	0D25	000D000A
***L**F	0A	25	000A
***N**L	0A	15	000A

DELIMITER = ***S**TD

Files added by BS2ZIP that include an extensible data field will use the delimiter specified within that field for separation. This delimiter can be displayed with SHOW-FILE-ATTRBIUTES INFO=*ALL. For files that were added without extensible data field (by third party utilities or BS2ZIP older than V21.0B10) will look for all of the delimiters associated with the encoding, e.g. if file is in EBCDIC, then BS2ZIP will look for 0D25, 25, 15.

DELIMITER = ***C**RLF / ***L**F / ***N**L
BS2ZIP will look for of the delimiters associated with the encoding of the file, e.g. if file is in EBCDIC, then BS2ZIP will look only for 25, if LF is specified.

DELIMITER = ***0D0**A / ***0A / *0D2**5 / ***2**5 / ***1**5 / ***000D**000A / ***000A**
BS2ZIP will ignore the encoding of the file and will look only for the specified line delimiter.

LOGGING = *MINIMUM / *MAXIMUM
Controls the amount of the message output.

LOGGING = *MINIMUM
Only error messages will be sent.

LOGGING =*MAXIMUM
All messages will be sent. Currently the [guaranteed] message SZP0122 is sent after each file extraction; further messages may be added in the future.

Notes

- If data encryption had been set using the MODIFY-ZIP-OPTIONS statement, encrypted files are decrypted again when they are extracted. The standard Zip 2.0 encryption used here is compatible with WinZip on Windows-based systems.
- Files extracted from a container created on the BS2000, are created with the same organization characteristics as the original file, except the padding factor and blockcontrol. The padding factor is the default DMS padding value. This implies that the size of extracted SAM and ISAM file can be different from the size of the original files.
- Files extracted from a container created in a foreign environment, are created
  - as SAM files with BUF-LEN=STD(16), provided that DATA-TYPE=*NOT-SPECIFIED or *CHARACTER,
  - as PAM files with BUF-LEN=STD(16) provided that DATA-TYPE=*BINARY and
  - as SAM files with REC-FORM=U provided that DATA-TYPE=*SAM-BINARY.
- K and NK disks
  - When a zipped file with PAMKEY block control has to be extracted on a NK disk, use the operand BLOCK-CONTROL-INFO=*IGNORE. The file will be converted into NK format. However, in case of SAM or ISAM files with records occupying all the available space in blocks, data truncation will occur. In this case, an error will be detected and the extract processing is aborted for the current file. The output file is erased.
  - To extract NK disk files (especially load modules), that have been added to a container, on a PAMKEY disk, the option BLOCK-CONTROL-INFO=*KEEP must be set.
- If the K2 key is pressed during EXTRACT-FILE statement, processing is interrupted with the query message `SZP0208`:
  - The user can simply continue processing.
  - The user can terminate processing and return to statement mode (//). The files which had not been added by the time the interruption occurred are not extracted. If required, they must be extracted again.
- To select files from an archive coming from open systems, take into account that the registered file names are case sensitive. So, use the c-string format to select filenames containing lower cases.

- Rules for naming extracted files:

  BS2000 files:
  (please refer to SDF rules for wildcard construction in the “Commands” manual [2])

FILE-NAME	TO-FILE	Registered file names	Resulting file names [2]
MYFILE1	*BY-SOURCE	MYFILE1	:ccid:$cuid.MYFILE1
*	*BY-SOURCE	MYFILE1 MYFILE2	:ccid:$cuid.MYFILE1 :ccid:$cuid.MYFILE2
MY*	EXT-*	MYFILE1 MYFILE2	:ccid:$cuid.EXT-FILE1 :ccid:$cuid.EXT-FILE2
MYFILE1	*BY-SOURCE	:XXXX:$UID.MYFILE1	No file found
:XXXX:$UID.	*BY-SOURCE	:XXXX:$UID.MYFILE1	:XXXX:$UID.MYFILE1
$UID.	*BY-SOURCE	$UID.MYFILE1	:ccid:$UID.MYFILE1

[2] where $cuid = current userid and :ccid: = catid of the userid

Not BS2000 files:

FILE-NAME	TO-FILE	Registered file names	Resulting file names [3]
MYFILE1	*BY-SOURCE	/temp/data/myfile.txt	No file found
*	*BY-SOURCE	/temp/data/myfile1.txt /temp/data/myfile2.txt	:ccid:$cuid.MYFILE1.TXT :ccid:$cuid.MYFILE2.TXT
'*myfile*'	EXT-*	/temp/data/myfile1.txt /temp/data/myfile2.txt	:ccid:$cuid.EXT-MYFILE1.TXT :ccid:$cuid.EXT-MYFILE2.TXT

[3]where $cuid = current userid and :ccid: = catid of the userid

3.2.6 MODIFY-ZIP-OPTIONS

This statement defines the defaults for the current BS2ZIP program run.

You can specify that BS2ZIP files should be encrypted when they are added and decrypted when they are extracted. Encryption is inapplicable to empty file as they are stored without it as they only require to store their metadata. BS2ZIP creates the keys which are required from the crypto password specified,

MODIFY-**ZIP-OPT**IONS

ENCRYPTION = ***UNCHA**NGED / ***NO** / ***Y**ES(...)

***Y**ES(...)

| **CRYPTO-PASS**WORD = <c-string 1..100 with-low> / <x-string 1..200> / ***SECRET**

| ,**CONFIRM-PASS**WORD = ***NOT-SPECIFIED** / <c-string 1..100 with-low> / <x-string 1..200> /

| ***SECRET**

ENCRYPTION = *UNCHANGED / *NO / *YES(...)
Specifies whether or not added files must be encrypted, and extracted files must be decrypted.

ENCRYPTION = *UNCHANGED
The current setting is retained. ENCRYPTION=*NO is set when BS2ZIP is started.

ENCRYPTION = *NO
Specifies that the encryption may not be used. If a password has been specified by a previous statement, the encryption keys will be removed.

ENCRYPTION = *YES(...)
Specifies that the encryption must be used, for adding files and for extracting encrypted files.

CRYPTO-PASSWORD = <c-string 1..100 with-low> / <x-string 1..200> / *SECRET Specifies the password to be used for encrypting and decrypting data.
The operand has the following special characteristics:

- The password entered is not logged.
- The input field is automatically blanked out in the guided dialog.
- If *SECRET or ^ is specified, in unguided dialog and in foreground procedures SDF provides a non-displaying entry field for concealed entry of the password.

CONFIRM-PASSWORD = *NOT-SPECIFIED / <c-string 1..100 with-low> / <x-string 1..200> / *SECRET
Permits the entry check for the crypto password. Entering the password twice is to avoid a password containing a typing error being assigned when password entry is nondisplaying.
If a value other than the default *NOT-SPECIFIED is entered, then is must be identical with the entry made for CRYPTO-PASSWORD, otherwise the statement is rejected. The operand has the following special characteristics:

- The password entered is not logged.
- The input field is automatically blanked out in the guided dialog.
- If *SECRET or ^ is specified, in unguided dialog and in foreground procedures SDF provides a non-displaying entry field for concealed entry of the password.

Note

As soon as the password is recognized, it is treated by BS2ZIP in order to store encryption keys in memory. The password itself is deleted from the memory.

3.2.7 OPEN-ZIP-CONTAINER

This statement opens a ZIP container. If a ZIP container had already been opened, it is closed again first.

By default the ZIP container is opened in read mode. If files are to be added to or deleted from it, it must be opened in write mode.

```
OPEN-ZIP-CONTAINER

CONTAINER = <filename 1..54 without-gen-vers> / *LINK(...)

    *LINK(...)

        |    LINK-NAME = <structured-name 1..8>

,MODE = *READ / *UPDATE(...)

    *UPDATE(...)

        |    STATE = *ANY / *NEW

,FORMAT = *STD / *WINZIP-COMPATIBLE / *BS2000
```

CONTAINER = <filename 1..54 without-gen-vers>
Name of the ZIP file to open.

CONTAINER = *LINK(...)
ZIP file to open is referenced by a link name.

LINK-NAME = <structured-name 1..8>
Link name associated to the ZIP file to open.

MODE = *READ
The ZIP file is opened in read mode. The file must exists.

MODE = *UPDATE(...)
The ZIP file is opened in write mode.

STATE = *ANY
If the specified ZIP file does not exists, it is created and opened in update mode, otherwise, it is opened in update mode.

STATE = *NEW
A new ZIP file is created and opened in update mode. If the ZIP file already exists, an error occurs and the statement is rejected.

FORMAT = *STD
If the container does not exist and must be created, i.e. when MODE=*UPDATE, the file is created in BS2000 format. If the container exists i.e. when MODE=*READ or
*UPDATE(STATE=*ANY), the file is opened in the format it has been created.

FORMAT = *BS2000
The ZIP container is opened in BS2000 format. ZIP containers in this format can only be used in a BS2000 environment.

FORMAT = *WINZIP-COMPATIBLE
The ZIP container is opened in WinZip-compatible format. ZIP containers in this format can be used in the open world as they are compatible with PKZIP V4.5.

Notes

- If a ZIP container was transferred to BS2000 using openFT < V11.0, it will have SAM file format with RECORD-FORMAT=U. Before this container is opened in BS2ZIP, it must be converted to PAM file format using the CONVERT-ZIP-CONTAINER statement. In the other direction a ZIP container must be converted from PAM to SAM file format with RECORD-FORMAT=U before it can be transferred to another system using openFT < V11.0. See “CONVERT-ZIP-CONTAINER”.
- An error message will be displayed when the file is not recognized as a valid ZIP file.
- A ZIP container created in WINZIP-COMPATIBLE format cannot be reopened in BS2000 format and a ZIP container created in BS2000 format cannot be reopened in WINZIP-COMPATIBLE format.
- A ZIP container created under Windows or Unix system can only be opened in WinZip compatible mode.
- GZIP archives can only be opened in MODE=*READ. Only display or extraction is thus possible. The container format has not to be specified. Only the default FORMAT=*STD is allowed.
- Opening a container while another one is currently opened leads to the close of the current one. The current one is also closed even if the open of the second one fails.
- The OPEN-ZIP-CONTAINER statement cannot be interrupted by K2.

- Maximum ZIP container extents reached.

  The user can influence the memory space allocation of a ZIP container by means of an appropriate primary and secondary allocation when the file is created using the CREATE-FILE command. If the file does not yet exist, BS2ZIP creates the ZIP container with an initial size of 1920 PAM pages (primary allocation), and a secondary allocation of 192 PAM pages.

  If the container is expected to be large, the user should define a higher secondary allocation. Otherwise there is a danger that because it is constantly being expanded the container will receive the maximum possible number of extents and can then not be expanded any more. A file created using BS2ZIP default values has a maximum size of around 120 MB.

  Example of estimation

```
DATAFILE1 100,000pp
DATAFILE2  50,000pp
Total =   150,000pp
```

  With a compression ratio of 60%, we can estimate the size of the ZIP container to 60000 PAM pages. You can estimate less than this value. So, BS2ZIP will extend the file if necessary.

  Execute the following CREATE-FILE command:

```
/CREATE-FILE MYCONT.ZIP,SUPPORT=*PUBLIC(SPACE=*RELA(60000,100))
/START-ZIP
//OPEN-ZIP-CONTAINER CONTAINER=MYCONT.ZIP,MODE=*UPDATE(STATE=*ANY)
//ADD DATAFILE*
//END
```

  In the reverse direction, the initial size of 1920 PAM pages can be excessive, even if it is temporary. To avoid this problem, the user is advised to make an estimation of the resulting size, and specify adequate Space parameters in the CREATE-FILE
  command. For very small libraries, the necessary primary space allocation can be as small as 6 PAM pages.
- If the available space is lower than 1920 PAM pages for a userid, it is necessary to create the container before opening it with BS2ZIP.
- The size of a container is variable, for the same contents, depending on the manipulations that have been done, including the REORGANIZE-ZIP-CONTAINER statement.

3.2.8 REORGANIZE-ZIP-CONTAINER

This statement reorganizes ZIP container by creating a another file and copying all files from the archives there. Deleted files and files that were interrupted during ADD-FILE are skipped. By default this statement creates archives with names in the format of BS2ZIP.REORG.YYYY-MM-DD.HHMMSS.TMP and if the reorganization succeeds the original ZIP container is deleted and the new one is renamed to the filename of the original archive. To preserve the original archive it is possible to specify the output file explicitly.

REORGANIZE-ZIP-CONTAINER

CONTAINER = ***STD** / <filename 1..54 without-gen-vers> / ***LINK**(...)

***LINK**(...)

| LINK-NAME = <structured-name 1..8>

,OUTPUT-**C**ONTAINER = ***S**TD / <filename 1..54 without-gen-vers> / ***L**INK(...)

***L**INK(...)

LINK-NAME **=** <structured-name 1..8>

CONTAINER = *STD
The opened ZIP container is reorganized. If no ZIP container has been opened yet, the statement is rejected.

CONTAINER = <filename 1..54 without-gen-vers>

Name of the ZIP container to be reorganized.

CONTAINER = *LINK(...)
ZIP container to be reorganized is referenced by a link name.

LINK-NAME = <structured-name_1..8>
Link name associated to the ZIP container to be reorganized.

OUTPUT-**C**ONTAINER **= *S**TD
The file is reorganized into an intermediate file under file name location BS2ZIP.REORG.YYYY-MM-DD.HHmmSS.TMP. The intermediate archive substitutes the original archive and gets deleted. The original file can be opened only in *UPDATE mode.

OUTPUT-**C**ONTAINER = <filename 1..54 without-gen-vers>
The file is reorganized into an intermediate file under the specified file name location. This file is not deleted on a successful end. The original file stays unorganized and is not required to be opened in *UPDATE mode.

OUTPUT-**C**ONTAINER**= *L**INK(...)
The file is reorganized into an intermediate file under the file name location referenced by a link name. This file is not deleted on a successful end. The original file stays unorganized and is not required to be opened in *UPDATE mode.

LINK-NAME = <structured-name_1..8>
Link name associated to the ZIP container to be reorganized.

Notes

- During the reorganization of a ZIP container, no other access to this ZIP container is permitted.
- If the ZIP container has already been opened by another task, it cannot be reorganized.

3.2.9 SHOW-FILE-ATTRIBUTES

ofncoding

This statement displays the list of the files included in the currently opened ZIP file.

This statement supports structured output in S variables.

SHOW-FILE-ATTRIBUTES

FILE-NAME = ***ALL** / ***P**ATH-**N**AME(...) / <composed-name 1..98 with-under with-wild(132)> / <c-string 1..1024 with-low>

***P**ATH-**N**AME**(...)**

| PATH=<c-string 1..1800 with-low>

,**INF**ORMATION = ***SUMM**ARY / ***ALL**

,**TEXT-OUTPUT** = ***SYSOUT** / ***SYSLST**(...) / ***NONE**

***SYSLST**(...)

| **SYSLST-NUM**BER = ***STD** / <integer 1..99>

,**STRUCTURE-OUTPUT** = ***SYSINF** / ***NONE** / <composed-name 1..255>(...)

<composed-name 1..255>(...)

| **WRITE-MODE** = ***REPLACE** / ***EXTEND**

FILE-NAME = *ALL
All the files included in the container are listed.

FILE-NAME = <composed-name 1..98 with-under with-wild(132)>
All the files matching the pattern are listed.

FILE-NAME = <c-string 1..1024 with-low>
All files which match the specified string (wildcards according to the SDF rules for wildcard selection (see SDF syntax in the “Commands” manual [2]) are permitted) are extracted from the ZIP container. Wildcards may be specified in the input string.
All the matching files are listed. Specification as a C string must be used if the container was created in a non-BS2000 system and the file names concerned do not comply with BS2000 syntax (e.g. upper/lower case).

FILE-NAME = *PATH-NAME(...)
The specified file will be shown. The operand will not interpret characters like forward slash, asterisk, square brackets and others as wildcards, but as part of the filename. Use this operand to show a single relative path name inside of the ZIP container.

INFORMATION = *SUMMARY
Only the name of the archived file plus its origin is displayed. The display ends with the message SZP0087, which shows the total number of files listed. See also “Layout of the output information”.

INFORMATION = *ALL
Complete information about the archived files is displayed. The display ends with the message SZP0087, which shows the total number of files listed. See also “Layout of the output information”.

TEXT-OUTPUT = *SYSOUT / *SYSLST(...) /*NONE
Specifies where the information is to be output (output as text string).

TEXT-OUTPUT = *SYSOUT
The information is output to the system file SYSOUT.

TEXT-OUTPUT = *NONE
No information is output except error messages.

TEXT-OUTPUT = *SYSLST(...)
The information is output to the system file SYSLST.

SYSLST-NUMBER = *STD / <integer 1..99>
Specifies whether the information is to be output to the system file SYSLST or to a SYSLST file from the set SYSLST01 through SYSLST99.
The default is *STD, i.e. output is directed to the system file SYSLST.

STRUCTURE-OUTPUT = *SYSINF / *NONE / <composed-name 1..255>(...)
Controls structured output in S variables (see also “Structured output in S variables”).

STRUCTURE-OUTPUT = *SYSINF
The structured output is directed to the S variable stream SYSINF. The information is accessible if SYSINF was previously assigned to an S variable using the ASSIGN-STREAM command (see “Command output in S variables” in the “Commands” manual [2]).

STRUCTURE-OUTPUT = *NONE
No structured output is to be generated.

STRUCTURE-OUTPUT = <composed-name 1..255>(...)
Name of the variable into which the structured output is to be made. This variable must bedeclared as a list variable:

```
/DECLARE-VARIABLE VAR-NAME=<var>(TYPE=*STRUCTURE),MULTIPLE-ELEMENTS=*LIST
```

WRITE-MODE = *REPLACE / *EXTEND
Specifies whether the list variable is to be overwritten or extended.
*REPLACE means that the list is overwritten.

Layout of the output information

*Layout with INFORMATION=*SUMMARY*

```
CURRENT CONTAINER : @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@             @@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  + BS2000 : @@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  + BS2000 : @@@
...
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  + BS2000 : @@@
%  SZP0087 'n' file(s) are matching your request
```

The first line displays the name of the currently opened ZIP container and its format: BS2 (= BS2000 format) or WIN (= WinZip compatible format). The file name and the BS2000 output field are displayed in one line for each selected file. Encrypted files are marked with a “+” in front of the BS2000 output field. File names which are longer than 64 characters are displayed in more than one line.

*Layout with INFORMATION=*ALL:*

```
CURRENT CONTAINER : @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@             @@@
--------------------------- FILE INFORMATION --------------------------------
FILENAME    : @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
              @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
              ...
              @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
BS2000      : @@@
MODIFIED    : YYYY-MM-DD HH:MM:SS
SIZE        : NNNNNNNNNNNNN
PACKED      : NNNNNNNNNNNNN
RATIO       : NN.N %
ENCRYPTED   : @@@
CCSNAME     : @@@@@@@@ (CURRENT: @@@@@@@@; ORIGIN: @@@@@@@@)
DELIMITER   : @@@@@@@@
------------------------------- COMMENTS ------------------------------------
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
...
%  SZP0087 'n' file(s) are matching your request
```

Fields description

CURRENT CONTAINER	is the name of the currently opened ZIP container and its format: BS2 for BS2000 format or WIN for WinZip compatible format.
FILENAME	is the name of the file included in the ZIP container. In case of BS2000 file, this name has the following format: [:cid:][$uid.]filename according to the TO-FILE operand of the ADD-FILE statement. It is always in uppercase. In case of non-BS2000 files, this name has the format with which it has been saved. For example, /dir1/dir2/Myfile.txt. It is case sensitive. Up to 64 characters of the file name are displayed in one line. When file names are longer than 64 characters, the remaining characters are displayed in continuation lines. Up to 1024 characters can be displayed.
BS2000	Indicates the origin of the file.
YES	means that the file has been included in the container by BS2ZIP program in a BS2000 system
NO	otherwise
MODIFIED	is date and time of the last file modification.

SIZE
: is the size in bytes of the original file.

  In case of Gzip archive this field will display "NOT AVAILABLE", because Gzip does not store the original file length.

PACKED
: is the size in bytes of the file after compression.

RATIO
: is the compression ratio. It is computed according to the formula:
(SIZE – PACKED) * 100 / SIZE.

  It can occur that the packed size is greater than the original size when no compression is required at ADD-FILE statement.
In this case, the RATIO is 0.

  In case of Gzip archive this field will display "NOT AVAILABLE", because Gzip does not store the original file length.

ENCRYPTED
: Specifies whether the file was encrypted when it was added to the container.

  YES
  : displays an encrypted file.

  NO
  : displays a not encrypted file.

CCSNAME
: displays the encoding name in which the file is stored in the archive, its type, and the type of encoding of the original file.

  This field is displayed only if the file was added by BS2ZIP V21.0B10 or newer. If the file was added by older versions of BS2ZIP or third-party archive managers, then the file does not have the extensible data field containing the current encoding name.

  The encoding name is an at maximum 8-byte long name of an encoding as defined in XHCS and used in the CODED-CHARACTER-SET attribute of a file on BS2000. Examples: ISO8859F, EDF041, UTF8.

  The encoding type refers to the category of encoding, which determines how the program behaves. There are 4 types: *EBCDIC, *ASCII, *UNICODE, and *UNKNOWN.

  The encoding name is printed first, followed by the current and original encoding types in parentheses. The original encoding name can be found in the CODED-CHAR-SET attribute of the file-info comment. Example:

```
CCSNAME    : ISO8859F (CURRENT: *ASCII  ; ORIGIN: *EBCDIC )
```

DELIMITER
: is the characters that are used as end-of-line delimiter in the file. This field is shown only if the file has BS2ZIP custom extra field and the file is a text file.

COMMENTS
: is the comments associated to the file in archive. In the case of files added to archive by BS2ZIP, the comments begin with a **`*BS2:`** string and then list the DMS properties of the original file as operands of a FILE macro (see the “DMS Macros” manual [4]). This list of DMS properties is referred as "file-info" in this manual.

Note

There are 4 types of CCSNAMEs:

*EBCDIC	*ASCII	*UNICODE	*UNKNOWN
EDF03IRV	ISO88591	UTF8	All CCSNAMEs that
EDF03DRV	ISO88592	UTFE	are not in this table
EDF04DRV	ISO88593	UTF16	and that XHCS could
EDF041	ISO88594		not identify during
EDF042	ISO88595		ADD-FILE
EDF043	ISO88597		
EDF044	ISO88599		
EDF045	ISO8859F		
EDF046	WCP1252P		
EDF047			
EDF049			
EDF04A			
EDF04B			
EDF04C			
EDF04D			
EDF04E			
EDF04F			
EEHCL2			
EEHCLC			
EEHCLC1			
EEHCLAA			
EEHCLG			

The container format is also available in the file attributes of the container in the field USER-INFO (Organization section):

`USER-INFO = BS2ZIP-B` for BS2000 format

`USER-INFO = BS2ZIP-W` for WinZip compatible

Structured output in S variables

The INFORMATION=*SUMMARY operand supplies only the S variables for the file name and the file origin (`FILENAME` and `BS2000`) with values. INFORMATION=*ALL causes all the S variables to be supplied with values.

Output information	Name of the S variable	T	Contents

Alternate index (ISAM file)	var(*LIST).ALTERNATE-INDEX	S	'' *YES
Block control information	var(*LIST).BLKCTRL	S	NO PAMKEY DATA2K DATA4K DATA
Buffer size	var(*LIST).BLKSIZE	S	'' <integer> STD(n)
Origin of the file	var(*LIST).BS2000	S	*YES *NO
Name of coded character set of the original file	var(*LIST).CODED-CHAR-SET	S	'' <ccs>
Type of coded character set of the original file	var(*LIST).ORIGIN-CCSTYPE	S	'' *ASCII *EBCDIC *UNICODE *UNKNOWN
New line delimiter	var(*LIST).DELIMITER	S	'' 0A 0D0A 000D000A 0D15 15
File encryption	var(*LIST).ENCRYPTED		*YES *NO
File structure (access method used to create the file)	var(*LIST).FCBTYPE	S	SAM ISAM PAM
File name; in the case of non-BS2000 files possibly as a path name	var(*LIST).FILENAME	S	<filename> <pathname 1..1024>
Indicator for the file type PLAM library	var(*LIST).FILETYPE	S	'' PLAM-LIB

Performance requirements for I/O operations	var(*LIST).IOPERF	S	'' HIGH STD VERY-HIGH
Operation(s) affected by the I/O performance requirements	var(*LIST).IOUSAGE	S	'' READ RDWRT WRITE
Length of the ISAM key	var(*LIST).KEYLEN	I	<integer>
Position of the ISAM key in the record	var(*LIST).KEYPOS	I	<integer>
Validity of last byte pointer (LBP)	var(*LIST).LAST-BYTE-VALID	B	FALSE TRUE
Length of the logical flag in the ISAM index	var(*LIST).LOGLEN	I	<integer>
Date of last file modification	var(*LIST).MODIFIED-DATE	S	<yyyy-mm-dd>
Time of last file modification	var(*LIST).MODIFIED-TIME	S	<hh:mm:ss>
Size of compressed file (byte)	var(*LIST).PACKED	I	<integer>
Compression ratio	var(*LIST).RATIO	S	'' nn.n
File record format	var(*LIST).RECFORM	S	F U V
File record length	var(*LIST).RECSIZE	I	<integer>
Secondary allocation for file extensions	var(*LIST).SEC-ALLOC	I	<integer>
Original file size (byte)	var(*LIST).SIZE	I	<integer>
Name of coded character set in which file is stored in archive	var(*LIST).STORED-CCSNAME	S	'' <ccs>
Type of coded character set in which the file is stored in archive	var(*LIST).STORED-CCSTYPE	S	'' *ASCII *EBCDIC *UNICODE *UNKNOWN
Length of the value flag in the ISAM index	var(*LIST).VALLEN	I	<integer>

Treatment of the value flag within a data or index block (for K-ISAM files)	var(*LIST).VALPROP	S	'' MAX MIN

Example

```
/START-ZIP-MANAGER

//OPEN-ZIP-CONTAINER MYCONT.ZIP

//SHOW-FILE-ATTRIBUTES

CURRENT CONTAINER : MYCONT.ZIP                                          WIN

$USRA.ADVENTURE.TXT                                          BS2000 : YES

%  SZP0087 '1' file(s) are matching your request

//SHOW-FILE-ATTRIBUTES *ALL,INFORMATION=ALL
CURRENT CONTAINER : MYCONT.ZIP                                          WIN

--------------------------- FILE INFORMATION -------------------------------

FILENAME   : $USRA.ADVENTURE.TXT
BS2000     : YES

MODIFIED   : 2011-06-11 18:49:32

SIZE       : 227454976
PACKED     : 35012077

RATIO      : 84.6 %

ENCRYPTED  : NO

CCSNAME    : ISO8859F (CURRENT: *ASCII  ; ORIGIN: *EBCDIC )

DELIMITER  : 0D0A
------------------------------ COMMENTS ----------------------------------

*BS2: ,FCBTYPE=SAM,BLKSIZE=(STD,16),RECFORM=V,IOPERF=STD,IOUSAGE=RDWRT,CODED-CH

AR-SET=EDF04F  ,BLKCTRL=DATA,SEC-ALLOC=16

%  SZP0087 '1' file(s) are matching your request
```

3.2.10 START-TRACE

This statement activates an internal trace.

START-TRACE
TRACE-FILE = ***STD** / <filename 1..54 without-gen-vers>

TRACE-FILE = *STD
The trace file name by default is SYSTRC.BS2ZIP.<yyyy-mm-dd>.<hhmmss> and is located under the userid that executes the BS2ZIP program.

TRACE-FILE = <filename 1..54 without-gen-vers>
A trace file with the user specified file name is created. If the file already exists, it is overwritten.

3.2.11 STOP-TRACE

This statement deactivates the internal trace. The trace file is closed.

STOP-TRACE

3.3 Examples

BS2000 example

```
/show-file-attr test.
%         3 :1OSN:$BS2ZIP.TEST.ISAM-STD1-PAMKEY-VN0-EDF03IRV-8-5
%        18 :1OSN:$BS2ZIP.TEST.ISAM2
%         6 :1OSN:$BS2ZIP.TEST.ISO88591
%         3 :1OSN:$BS2ZIP.TEST.NKEY
%        12 :1OSN:$BS2ZIP.TEST.NKFILE
%        12 :1OSN:$BS2ZIP.TEST.PAM
%        51 :1OSN:$BS2ZIP.TEST.PAM.16VP-32768
%        36 :1OSN:$BS2ZIP.TEST.PAM16-MAX.2
%        21 :1OSN:$BS2ZIP.TEST.PAM2
%        48 :1OSN:$BS2ZIP.TEST.RECFORMU
%       192 :1OSN:$BS2ZIP.TEST.SAM.16VP-32768
%        18 :1OSN:$BS2ZIP.TEST.SAM1-MAX


/start-zip-manager

%  SZPLOAD Program 'BS2ZIP',Version 'V01.2G15' of '2014-09-15' loaded

%  SZPCOPY Copyright (C) 2014 Fujitsu Technology Solutions GmbH All Rights

Reserved

//open-zip-container container=mycont,mode=*update(state=*new)

//add-file test.

//show-file-attr
CURRENT CONTAINER : MYCONT

TEST.ISAM-STD1-PAMKEY-VN0-EDF03IRV-8-5                          BS2000 : YES
TEST.ISAM2                                                      BS2000 : YES
TEST.ISO88591                                                   BS2000 : YES
TEST.NKEY                                                       BS2000 : YES
TEST.NKFILE                                                     BS2000 :
YES
TEST.PAM                                                        BS2000 : YES
TEST.PAM.16VP-32768                                             BS2000 : YES
TEST.PAM16-MAX.2                                                BS2000 :
YES
TEST.PAM2                                                       BS2000 : YES
TEST.RECFORMU                                                   BS2000 : YES
TEST.SAM.16VP-32768                                             BS2000 :
YES
TEST.SAM1-MAX                                                   BS2000 :
YES

%  SZP0087 '12' file(s) are matching your request
```

```
//show-file-attr file-name=test.isam-std.*,inf=*all

CURRENT CONTAINER : MYCONT

 -------------------------- FILE INFORMATION ------------------------------
 FILENAME   : TEST.ISAM-STD1-PAMKEY-VN0-EDF03IRV-8-5

 BS2000     : YES

 MODIFIED   : 2014-11-25 10:21:22
 SIZE       : 231

 PACKED     : 56

 RATIO      : 75.8 %
 ENCRYPTED  : NO

 ------------------------------ COMMENTS ---------------------------------

 *BS2: ,FCBTYPE=ISAM,BLKSIZE=(STD,1),KEYPOS=5,KEYLEN=8,RECFORM=(V,N),
IOPERF=STD,IOUSAGE=RDWRT,CODED-CHAR-SET=EDF03IRV,BLKCTRL=PAMKEY

%  SZP0087 '1' file(s) are matching your request

//extract-files file-name=test.pam.*,to-file=ext-test.pam.*
//end

/show-file-attr ext-*

%        51 :1OSN:$BS2ZIP.EXT-TEST.PAM.16VP-32768
%        33 :1OSN:$BS2ZIP.EXT-TEST.PAM16-MAX.2
%        21 :1OSN:$BS2ZIP.EXT-TEST.PAM2
```

WinZip example

```
/start-ftp

.

.

.

ftp> dir
200 PORT command successful.

150 Opening ASCII mode data connection for /bin/ls.

05-15-14 03:19PM               978691 server_1.zip
226 Transfer complete.

53 bytes received in 0.08 seconds (0.62 Kbytes/s)

ftp> bin
200 Type set to I.

ftp> get server_1.zip SERVER-1.PC.ZIP
```

```
200 PORT command successful.

150 Opening BINARY mode data connection for server_1.zip(978691 bytes).

226 Transfer complete.

978691 bytes received in 6.90 seconds (1.4e+02 Kbytes/s)

ftp> bye

221

%  CCM0998 CPU TIME USED: 0.0925 SECONDS

/show-file-attr server-1.pc.zip,inf=all
%0000000480 :2OSG:$USER1.SERVER-1.PC.ZIP

%  ------------------------------
HISTORY      ------------------------------

%  CRE-DATE   = 2014-05-15  ACC-DATE   = 2014-05-15  CHANG-DATE = 2014-05-15
%  CRE-TIME   =   15:46:32  ACC-TIME   =   15:47:45  CHANG-TIME =   15:46:39

%  ACC-COUNT  = 3           S-ALLO-NUM = 0

%  ------------------------------
SECURITY     ------------------------------
%  READ-PASS  = NONE        WRITE-PASS = NONE        EXEC-PASS  = NONE

%  USER-ACC   = OWNER-ONLY  ACCESS     = WRITE       ACL        = NO

%  AUDIT      = NONE        FREE-DEL-D = *NONE       EXPIR-DATE = 2014-05-15
%  DESTROY    = NO         FREE-DEL-T = *NONE       EXPIR-TIME =   00:00:00

%  SP-REL-LOCK= NO          ENCRYPTION = *NONE

%  ------------------------------
BACKUP       ------------------------------
%  BACK-CLASS = A          SAVED-PAG  = COMPL-FILE  VERSION    = 1

%  MIGRATE    = ALLOWED

%  ------------------------------ ORGANIZATION
----------------------------
%  FILE-STRUC = PAM         BUF-LEN    = STD(1)      BLK-CONTR  = PAMKEY

%  IO(USAGE)  = READ-WRITE  IO(PERF)   = STD        DISK-WRITE = IMMEDIATE

%  AVAIL      = *STD
%  WORK-FILE  = *NO         F-PREFORM  = *K         S0-MIGR    = *ALLOWED

%  ------------------------------
ALLOCATION   ------------------------------

%  SUPPORT    = PUB         S-ALLOC    = 16          HIGH-US-PA = 478
%  EXTENTS     VOLUME     DEVICE-TYPE      EXTENTS     VOLUME     DEVICE-TYPE

%      1       GVS2.5       D3435
```

```
%  NUM-OF-EXT = 1
%:2OSG: PUBLIC:      1 FILE  RES=       480 FRE=        2 REL=        0
PAGES
```

/**start-zip**

```
%  SZPLOAD Program 'BS2ZIP',Version 'V01.2G15' of '2014-09-15' loaded
%  SZPCOPY Copyright (C) 2014 Fujitsu Technology Solutions GmbH All Rights
Reserved
```

//**open-zip-container server-1.pc.zip**

//**show-file-attr '*\/<ADD,Glos,Ind>*.htm'**

```
CURRENT CONTAINER : SERVER-1.PC.ZIP
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/test_kdo_1/ADD_CJC-ACTION.htm
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/test_kdo_1/ADD_FILE_LINK__TFT_Eintr.htm
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/test_kdo_2/ADD_CJC_ACTION.htm
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/test_kdo_2/ADD_FILE_LINK.htm
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/Glossary.htm
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Skin/  BS2000 : NO
Index.htm
%  SZP0087 '6' file(s) are matching your request
```

//**show-file-attr *ADD/CJC***

```
CURRENT CONTAINER : SERVER-1.PC.ZIP
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/test_kdo_1/ADD_CJC-ACTION.htm
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/test_kdo_2/ADD_CJC_ACTION.htm
%  SZP0087 '2' file(s) are matching your request
```

//**show-file-attr '*.css'**

```
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/Resources/Stylesheets/test_kdo_1.css
```

```
server_1/osd_allg/projekte/test_kdo_1/Output/new/MyWebHelp/Conte  BS2000 : NO
nt/SkinSupport/MadCap.css

%  SZP0087 '2' file(s) are matching your request

//extract-file file-name=*ADD/CJC*,to-file=webhelp.*.copy
%  SZP0090 Warning.  File name 'WEBHELP.ADD_CJC-ACTION.HTM.COPY' is not

BS2000 compliant.  The file will be extracted under the name

'FILE0001.20140515.155645'

%  SZP0090 Warning.  File name 'WEBHELP.ADD_CJC_ACTION.HTM.COPY' is not

BS2000 compliant.  The file will be extracted under the name

'FILE0002.20140515.155645'

//extract-file file-name='*.css',to-file=webhelp.*.copy

%  SZP0090 Warning.  File name 'WEBHELP.TEST_KDO_1.CSS.COPY' is not BS2000

compliant.  The file will be extracted under the name
'FILE0001.20140515.155759'

//end

/show-file-attr <file,webhelp>*

%        33 :2OSG:$USER1.FILE0001.20140515.155645
%        48 :2OSG:$USER1.FILE0001.20140515.155759
%        33 :2OSG:$USER1.FILE0002.20140515.155645
%        33 :2OSG:$USER1.WEBHELP.MADCAP.CSS.COPY

%:2OSG: PUBLIC:      4 FILES RES=       147 FRE=       19 REL=        0
PAGES
```

4 Interoperability

4.1 Windows interoperability

Windows programs supporting the ZIP format (WinZip) are able to read BS2ZIP containers provided those BS2000 containers have been created in WinZip compatible mode.

BS2ZIP is able to read Windows ZIP archives provided they are opened and BS2ZIP is able to read them in compatible mode.

The transfer of such containers must be performed in binary format. The following shows an example of transfer.

Under DOS command prompt:

```
D:\>ftp
ftp>open 999.999.999.999
connected to 999.999.999.999
user : bs2zip
password : *******
account : xxxx
ftp>binary
ftp>quote ftyp binary
ftp>get bs2000.zip
ftp>put pc.zip
ftp>bye
```

`D:\bs2000.zip` can be opened by WinZip and `$BS2ZIP.PC.ZIP` can be opened by BS2ZIP.

> The length of the SAM and ISAM file records is limited by the buffer length of the file (BUFFER-LENGTH). This does not apply for records of Windows text files. To minimize the risk of too long records, BS2ZIP always specify a buffer length *STD(16) which corresponds to 32 KB - 8 Byte. This must be taken into account by the applications reading such files.
>
> If longer records are found, they are split in as many records as necessary.

Transferring ZIP files with openFT

When transferring ZIP files from/to Windows with openFT< V11.0, the ZIP files must be converted into the correct file format, either before transfer (transfer to Windows) or after transfer (transfer from Windows).

Indeed,

1. BS2ZIP only manages PAM files withBUFFER-LENGTH=STD(x)
2. PAM files may not be transferred to Windows with openFT< V11.0
3. the files must be transferred in binary mode (mandatory for zipped files)
4. the resulting file transferred (from Windows) with openFT< V11.0 is a SAM file with BUFFER-LENGTH=STD(x) and RECORD-FORMAT=U

SAM/PAM conversion

You can execute conversion in BS2ZIP using the CONVERT-ZIP-CONTAINER statement. See the description of “CONVERT-ZIP-CONTAINER”.

For compatibility reasons the converter can also be called using the /START-SAM-PAM-CONVERTER command:

You are prompted to specify the input file (first parameter) and the output file (second parameter). The converter converts the input file as follows:

- a PAM file with BUF-LEN=STD(x) into a SAM file with BUF-LEN=STD(8)
- a SAM file with BUF-LEN=STD(x) into a PAM file with BUF-LEN=STD(16)

4.2 Unix system interoperability

GZIP or GZ Unix system tools generate GZIP archives. Those archives must be transferred in binary to the BS2000 and then can be read by BS2ZIP. Those archives cannot be modified with BS2ZIP.

By default, BS2ZIP extracts the archived file as a SAM file with BUF-LEN=STD(16). It is assumed that the line separator is a Line Feed (LF). Files can be also be extracted in binary PAM files with BUF-LEN=STD(16) are thus created.

> ¡ Only gz files are supported; tar.gz files are not supported.

4.3 Linux interoperability

Archives created by ZIP Linux tool are readable by BS2ZIP. Those archives must be transferred in binary to the BS2000. By default, BS2ZIP extracts the archived file as a SAM file with BUF-LEN=STD(16). It is assumed that the line separator is a Carriage Return / Line Feed (CRLF). Files can be also be extracted in binary. PAM files with BUF-LEN=STD(16) are thus created.

On the other hand, BS2ZIP containers built in compatible mode can be transferred in binary to a Linux platform and then be exploited by the UNZIP Linux tool.

4.4 Character encoding

Here and everywhere else in this document, where UTF16 is mentioned, UTF16 big endian is implied. Little endian variant of UTF16 is not supported as it is not supported by XHCS. When files in UTF16 are added/extracted to/from archives, BS2ZIP does not add BOM if it is absent in the file and it does not remove it if it is present.
Both Zip and Gzip archives created on open platforms do not have information about the encoding of files inside of them. This is why by default before extracting a file, added to archive by different archive manager, BS2ZIP scans the first 32 kiB of it to determine the most likely encoding between ISO8859F, WCP1252, UTF8 and UTF16. To guaranty a correct encoding and skip the scan use EXTRACT-FILE with CHAR-ENCODING = *BY-PARAMETERS(...).

Files added by BS2ZIP to winzip compatible archives with DATA-TYPE = *NOT-SPECIFIED or *CHARACTER have CRLF in corresponding encoding as line delimiter.

5 Application Program Interface

BS2ZIP can be used as sub-program in TU applications. To that purpose, an API and a run time module are provided in the SYSLNK.BS2ZIP.<version>.RTE library:

- The C++ header file SZPZIP.H: API of BS2ZIP
- The C++ header file SZPZOUT.H: layout of the output buffers for the *ListFiles* function
- The run time module BS2ZIPPR

5.1 SZPZIP.H

```
#ifndef __SzpZip_h__
#define __SzpZip_h__
/*****************************************************************
 * BEGIN-INTERFACE  SZPZIP.H
 *
 *  TITLE   (/ zip program interface  /)
 *  Classes        CSzpZip
 *  NAME           szpzip.h
 *  DOMAIN         BS2ZIP
 *  LANGUAGE       CXX
 *
 *  Copyright (C) 2021 Fujitsu Technology Solutions GmbH
 *                     All rights reserved
 *
 *  COMPILATION-SCOPE RESTRICTED
 *  INTERFACE-TYPE    CALL
 *  RUN-CONTEXT       TU
 *
 *  VERSION        140
 *  Author         (/ V.Malets,I.Liatkouski IBA-BS2-CORE /)
 *  Date           2022-08-31
 *  Update         (/ BS2ZIP 21.0A20
 *                 #CORR PM#=A0616124
 *                 Added new operand *PATH-NAME
 *                 for ADD-FILE statement.
 *                 List of changes:
```

```
 *                - BS2UTIL:
 *                Exchanging blanks in constructor with
 *                non-blank character '59'X and vice
 *                versa to avoid truncation depending on
 *                blank location in CMDWCOI macro.
 *                - SZPADDP:
 *                New pathname property related to a new
 *                *PATH-NAME operand
 *                - SZPZIP:
 *                Boolean pathname-field added to check
 *                if '*PATH-NAME' parameter was specified
 *                in ADD-FILE statement. If it is, then
 *                check out if wildcard syntax not used
 *                and further processing of to-file
 *                operand as a new filename. New messages
 *                SZP0124-SZP0127
 *                - SZPSTMT:
 *                Flag setting depending on *PATH-NAME
 *                parameter usage.
 *                - ZIPARCH:
 *                FindFile method method now accepts
 *                *PATH-NAME operand
 *                /)
 *
 * END-INTERFACE
 ************************************************************************/
/*
 *  VERSION        139
 *  CRDATE         2021-08-19
 *  AUTHOR         (/ NDC BS2 /)
 *  UPDATE         (/ BS2ZIP 21.0
 *                 Character conversion using
 *                 *select operand
 *                 /)
 *  VERSION        135
 *  Date           2018-07-03
 *  Author         (/ NDC BS2 /)
 *  Update         (/ BS2ZIP v1.2K
 *                 Correction of PM # A0612521
 *                 Fixed the issue of visibility deleted files
 *                 via 7-zip in windows-compatible archive /)
 *
 *  VERSION        134
 *  Date           2018-02-23
 *  Author         C. Kasap
 *  Update         BS2ZIP v1.2J
 *                 Correction of PM # A0611565
 *                 Fixed the issue where adding files to the zip container
```

```
*                failed due to trace file being included
*
*  VERSION        133
*  CRDATE         2014-04-08
*  AUTHOR         (/ J. Beaume, OSL41 /)
*  UPDATE         (/ BS2ZIP v1.2G10 /)
*
*  VERSION        132
*  Date           2013-02-01
*  Author         P. Louis - OSL EPS
*  Update         BS2ZIP v1.2E05
*
*  VERSION        130
*  Date           2011-05-18
*  Author         J. Beaume - OSL EPS
*  Update         BS2ZIP v1.2D05
*
*  VERSION        129
*  Date           2009-09-30
*  Author         P. Louis - OSL EPS
*  Update         BS2ZIP v1.2C05
*
*  VERSION        128
*  Date           2008-11-27
*  Author         P. Louis - OSL EPS
*  Update         BS2ZIP v1.2B05
*                 Introduction CpuExhauster (A0571814)
*
*  VERSION        126
*  Date           2008-10-14
*  Author         Ph/Dumont- OSL EPS
*  Update         BS2ZIP v1.2A25
*                 Introduction CpuExhauster (A0570787)
*
*  VERSION        121
*  Date           2008-04-18
*  Author         P.Louis  - OSL EPS
*  Update         BS2ZIP v1.2A10
*
*  VERSION        118
*  Date           2007-12-01
*  Author         P.Louis  - OSL EPS
*  Update         BS2ZIP v1.2A
*
*  VERSION        116
*  Date           2006-12-01
*  Author         Ph.Dumont  - OSL EPS
*  Update         BS2ZIP v1.1C
```

```
 *
 *  VERSION        115
 *  Date           2005-06-23
 *  Author         L. Tambour - OSL EPS
 *  Update         BS2ZIP v1.1B
 *
 *  VERSION        110
 *  Date           2004-09-30
 *  Author         L. Tambour - OSL EPS
 *  Update         BS2ZIP v1.1A
 *
 *  VERSION        100
 *  Date           2003-06-02
 *  Author         J. Beaume - OSL EPS
 *  Update         BS2ZIP v1.0B
 ****************************************************************/
#include "szpzout.h"
// SzpZip return codes:
#define CSZPZIP_MULT_FILE_REJECTION   127    //140
#define CSZPZIP_LEAD_SLASH_REJECTION  126    //140
#define CSZPZIP_DRIVE_REJECTION       125    //140
#define CSZPZIP_BACKSLASH_REJECTION   124    //140
#define CSZPZIP_LIB_EXCEPT_ERROR      114    //133
#define CSZPZIP_FILE_EXCEPT_ERROR     113    //133
#define CSZPZIP_LIB_SELECTION_ERROR   112    //133
#define CSZPZIP_FILE_SELECTION_ERROR  111    //133
#define CSZPZIP_DELETE_ORILIBEL_ERROR 110
#define CSZPZIP_DELETE_ORIGFILE_DMS   109
#define CSZPZIP_DELETE_ORIGFILE_ERROR 108
#define CSZPZIP_LMS_ERROR             101
#define CSZPZIP_PARAMETER_ERROR       100
#define CSZPZIP_INCOHERENT_FORMAT      99
#define CSZPZIP_OPEN_ERROR             98
#define CSZPZIP_CONTAINER_EXISTS       97
#define CSZPZIP_NO_CONTAINER_EXISTS    96
#define CSZPZIP_LINKNAME               95
#define CSZPZIP_NO_FILE                94
#define CSZPZIP_FILE_EXISTS            93
#define CSZPZIP_NO_FILE_EXISTS         92
#define CSZPZIP_INTERNAL_ERROR         91
#define CSZPZIP_RSV1                   90
#define CSZPZIP_EXTRACT_ERROR          89
#define CSZPZIP_CATALOG                88
#define CSZPZIP_NO_CONTAINER_OPENED    86
#define CSZPZIP_READMODE               85
#define CSZPZIP_ALREADY_ZIPPED         84
#define CSZPZIP_ALREADY_ZIPPED_WLDC    83
#define CSZPZIP_ADD_ERROR              82
```

```
#define CSZPZIP_ILLEGAL_NEW_NAME       81
#define CSZPZIP_ILLEGAL_LINK           80
#define CSZPZIP_INVALID_RENAMING       79
#define CSZPZIP_IMPOSSIBLE_RENAMING    39
#define CSZPZIP_REORG_ERROR            78
#define CSZPZIP_DELETE_ERROR           77
#define CSZPZIP_INCON_FORMAT_ERROR     76
#define CSZPZIP_OUCON_DMS_ERROR        75
#define CSZPZIP_INCON_DMS_ERROR        74
#define CSZPZIP_OUCON_EXIST_ERROR      73
#define CSZPZIP_OUCON_NOEXIST_ERROR    72
#define CSZPZIP_INCON_NOEXIST_ERROR    71
#define CSZPZIP_OUCON_LINK_ERROR       70
#define CSZPZIP_INCON_LINK_ERROR       69
#define CSZPZIP_OUCON_FLINK_ERROR      68
#define CSZPZIP_INCON_FLINK_ERROR      67
#define CSZPZIP_INCON_OPEN_ERROR       66
#define CSZPZIP_PSWORDCHECK_ERROR      42
#define CSZPZIP_PSWORDNOTCOR_ERROR     41
#define CSZPZIP_PSWORDNOTGIV_ERROR     40
#define CSZPZIP_DATATYPE_BS2000        37
#define CSZPZIP_DMS_ERROR              31
#define CSZPZIP_INT_ERROR              19
#define CSZPZIP_STD_NAME                2
#define CSZPZIP_SHORT_STD_NAME          1
// Message id
#define MSG_K2                  "SZP0207"
#define MSG_MULT_FILE_REJECTION "SZP0127"  //140
#define MSG_SLASH_REJECTION     "SZP0126"  //140
#define MSG_DRIVE_REJECTION     "SZP0125"  //140
#define MSG_BACKSLASH_REJECTION "SZP0124"  //140
#define MSG_LIB_EXCEPT_ERROR    "SZP0114"  //133
#define MSG_FILE_EXCEPT_ERROR   "SZP0113" //133
#define MSG_LIB_SELECTION_ERROR "SZP0112"  //133
#define MSG_FILE_SELECTION_ERROR "SZP0111" //133
#define MSG_DELETE_ORILEL_FILE  "SZP0110"  //133
#define MSG_DELETE_ORIDMS_FILE  "SZP0109"
#define MSG_DELETE_ORIGIN_FILE  "SZP0108"
#define MSG_LMS_ERROR           "SZP0101"
#define MSG_PARAMETER_ERROR     "SZP0100"
#define MSG_INCOHERENT_FORMAT   "SZP0099"
#define MSG_OPEN_ERROR          "SZP0098"
#define MSG_CONTAINER_EXISTS    "SZP0097"
#define MSG_NO_CONTAINER_EXISTS "SZP0096"
#define MSG_LINKNAME            "SZP0095"
#define MSG_NO_FILE             "SZP0094"
#define MSG_FILE_EXISTS         "SZP0093"
#define MSG_NO_FILE_EXISTS      "SZP0092"
```

```
 #define MSG_INTERNAL_ERROR      "SZP0091"
#define MSG_WARNING1            "SZP0090"
#define MSG_EXTRACT_ERROR       "SZP0089"
#define MSG_CATALOG             "SZP0088"
#define MSG_NO_CONTAINER_OPENED "SZP0086"
#define MSG_READMODE            "SZP0085"
#define MSG_ALREADY_ZIPPED      "SZP0084"
#define MSG_ALREADY_ZIPPED_WLDC "SZP0083"
#define MSG_ADD_ERROR           "SZP0082"
#define MSG_ILLEGAL_NEW_NAME    "SZP0081"
#define MSG_ILLEGAL_LINK        "SZP0080"
#define MSG_INVALID_RENAMING    "SZP0079"
#define MSG_IMPOSSIBLE_RENAMING "SZP0039"
#define MSG_REORG_ERROR         "SZP0078"
#define MSG_DELETE_ERROR        "SZP0077"
#define MSG_INCON_FORMAT_ERROR  "SZP0076"
#define MSG_OUCON_DMS_ERROR     "SZP0075"
#define MSG_INCON_DMS_ERROR     "SZP0074"
#define MSG_OUCON_EXIST_ERROR   "SZP0073"
#define MSG_OUCON_NOEXIST_ERROR "SZP0072"
#define MSG_INCON_NOEXIST_ERROR "SZP0071"
#define MSG_OUCON_LINK_ERROR    "SZP0070"
#define MSG_INCON_LINK_ERROR    "SZP0069"
#define MSG_OUCON_FLINK_ERROR   "SZP0068"
#define MSG_INCON_FLINK_ERROR   "SZP0067"
#define MSG_INCON_OPEN_ERROR    "SZP0066"
#define MSG_PSWORDCHECK_ERROR   "SZP0042"
#define MSG_PSWORDNOTCOR_ERROR  "SZP0041"
#define MSG_PSWORDNOTGIV_ERROR  "SZP0040"
#define MSG_DATATYPE_BS2000     "SZP0037"
#define MSG_DMS_ERROR           "SZP0031"
#define MSG_INT_ERROR           "SZP0019"
/**
This class is the C++ api for managing a BS2ZIP container file.
This class is able to handle the K2 but not to intercept it.
**/
class CSzpZip {
public:
   /**
   **/
   CSzpZip();
   /**
   **/
   CSzpZip(bool bOutputMsg);
   /**
   **/
   virtual ~CSzpZip();
   enum szpOpenMode {
```

```
      read,
      updateAny,
      updateNew
   };
   enum szpFormat {
      defaut,
      compatible,
      bs2000
   };
   enum szpConvertMode {
      std,
      none,
      ascii,
      ebcdic,
      byParameters
   };
   enum szpWriteMode {
      create,
      replace,
      any
   };
   enum szpLink {
      no,
      yes
   };
   enum szpDataType {
      notSpecified,
      character,
      binary,
      sambinary
   };
   enum szpBlkCtrlInfo {
      keep,
      ignore
   };
   enum szpLevel {
      noCompression = 0,
      bestSpeed = 1,
      bestCompression = 9,
      defaultCompression = -1
   };
   enum szpInfo  {
      infoNone,
      infoAll,
      infoSummary
   };
// 132
   enum szpDelete {
```

```
   DeleteOption_No,
   DeleteOption_Yes
};
enum szpTofileOption {
    TofileOptionVersStd_TypeStd = 0,
    TofileOptionVersStd_TypeYes = 1,
    TofileOptionVersStd_TypeNo = 2,
    TofileOptionVersYes_TypeStd = 10,
    TofileOptionVersYes_TypeYes = 11,
    TofileOptionVersYes_TypeNo = 12,
    TofileOptionVersNo_TypeStd = 20,
    TofileOptionVersNo_TypeYes = 21,
    TofileOptionVersNo_TypeNo = 22
};
/**
Open a zip container.
It returns 0 if ok.
char *pContainer = name of the zip container file
this file name must be valid
if "link=linkname", the container is opened
using this link name
int iOpenMode    = read, update, create
int iFormat      = bs2000 or compatible
**/
int OpenZip(char *pContainer, int iOpenMode, int iFormat);
/**
Close the zip container and release eventually file link.
It returns 0 if ok.
**/
int CloseZip();
/**
Add files in the zip container.  The filename may contain wildcards.
By this way several files can be zipped in a single operation.  Eventual
data conversion is required as well as the compression level.
Files are compressed one by one.  If an error occurred, the data already
written in the container remain in it but the entry is not visible.  If
several files have to be written, the processing goes on even in case of
error on a previous file.  The same behavior is implemented if K2 is handled
during the processing of a file but if there are still files to be included,
those ones are not processed.
To keep the container coherent, the central header is rewritten after each
added files and then reread to rebuild the structure in memory.
It returns 0 if ok.
V01.0B: introduction of the parameter data-type
This operand is only significant for adding sam files in compatible format,
otherwise, it is ignored. In char mode, each record of a sam file is
interpreted as a line.  In binary mode, data are interpreted as a simple
byte stream.  Such a file is extracted as a pam std 16.
```

```
   char *pFileNames = file name (incl. wild cards ev.)
   int iConvertMode = convert mode
   int iLevel       = compression level
   int iDataType    = data type (not specified, character, binary)
   char *pToFiles   = target construction file name
   int iDeleteOption = source file delete option                          132
   **/
   int AddFiles(char *pFileNames, int iConvertMode, int iLevel, int iDataType, char
*pToFiles, int iDeleteOption);
   /**
   Add files in the zip container.  Old format for program compatibility.
   **/
   int AddFiles(char *pFileNames, int iConvertMode, int iLevel, int iDataType, char
*pToFiles);
   /**
   Add PLAM elements in the zip container.  The element names, version, base and type
   may contain wildcards.  Library name may not.
   By this way several elements can be zipped in a single operation.  Eventual
   data conversion is required as well as the compression level.
   Elements are compressed one by one.  If an error occurred, the data already
   written in the container remain in it but the entry is not visible.  If
   several elements have to be written, the processing goes on even in case of
   error on a previous file.  The same behavior is implemented if K2 is handled
   during the processing of a file but if there are still files to be included,
   those ones are not processed.
   To keep the container coherent, the central header is rewritten after each
   added elements and then reread to rebuild the structure in memory.
   It returns 0 if ok.
   V01.0B: introduction of the parameter data-type
   This operand is only significant for adding sam files in compatible format,
   otherwise, it is ignored. In char mode, each record of a sam file is
   interpreted as a line.  In binary mode, data are interpreted as a simple
   byte stream.  Such a file is extracted as a pam std 16.
   char *pLibName = library name (excl. wild cards)
   char *pElements = element name selector (incl. wild cards ev.)
   char *pVersion = element version selector (incl. wild cards ev.)
   char *pType = element type selector (incl. wild cards ev.)
   char *pBase = version base selector
   int iConvertMode = convert mode
   int iLevel       = compression level
   int iDataType    = data type (not specified, character, binary)
   char *pToFiles   = target construction file name
   int iTofileOption = specifies if libr element version and/or type are saved in the name
   int iDeleteOption = library element delete option                      132
   **/
   int AddLibElements(char *pLibName, char *pElements, char *pVersion,
          char *pType,char *pBase, int iConvertMode, int iLevel, int iDataType,
          char *pToFiles, int iTofileOption, int iDeleteOption);
   /**
   Add PLAM elements in the zip container.  Old format for program compatibility.
```

```
  **/
   int AddLibElements(char *pLibName, char *pElements, char *pVersion,
          char *pType,char *pBase, int iConvertMode, int iLevel, int iDataType,
          char *pToFiles, int iTofileOption);
   /**
   Extract one or several files selected by the pFileNames patterns into
   BS2000 files named according the rule specified by the pToFiles pattern.
   If an error occurred during the extract of a file, the extract of this
   file is interrupted but the other files are processed.
   If K2 is handled, the current file process is stopped, the output file
   is deleted and the other extract files are not processed if any.
   char *pFileNames = file pattern used to select the files that must
   be extracted from the container
   char *pToFiles   = pattern specifying the output name of the extracted files
   int iWriteMode   = new (default) or replace
   int iDataType    = data type of the file (not specified (default), char or
   binary)
   int iConvertMode = convert mode (std (default), none, ascii or ebcdic)
   int iBlkCtrlInfo = specifies if the original block control must be used
   or not
   **/
   int ExtractFiles(char *pFileNames, char *pToFiles,
   int iWriteMode = 0, int iDataType = 0, int iConvertMode = 0,
   int iBlkCtrlInfo = 0, bool pname = false);
   /**
   Returns the information about the zipped files selected by the pFileNames
   pattern that may contain wild cards.  Information summary or full is possible.
   The layout of the output is described in the header file szpzout.h.
   Data are returned in a buffer requested by this method.  Its address and size
   are returned to the caller who is in charge to release it after use.
   The number of matching files is returned.  -1 is returned in case of error.
   char *pFileNames = file selection pattern
   char **pBuf      = address where the output buffer address is returned
   int *iSize       = address of an int where the output buffer size is returned
   int iInfo        = information type selection (infoAll default, infoSummary)
   **/
   int ListFiles(char *pFileNames, char **pBuf, int *iSize = 0, int iInfo = infoAll, bool
ListFirst = true, bool pname = false);
     /**
   Delete files from a zip container.  The filename may contain wildcards.
   By this way several files can be deleted in a single operation.
   If several files have to be deleted, the processing goes on even in case of
   error on a previous file.  The same behavior is implemented if K2 is handled
   during the processing of a file but if there are still files to be deleted,
   those ones are not processed.

   It returns 0 if ok.
   char *pFileNames = file name (incl. wild cards ev.)
   **/
```

```
      int DeleteFiles(char *pFileNames, bool pname);
   /**
   CONVERT  ****************************************

   It returns 0 if ok.
   char *pFromContainer = Output file name
   char *pToContainer = Input file name
   int iHuser = Home userid length
   int iHcat  = Home catid length
   int iWriteMode   = new (default) or replace
   **/
   int CnvZip(char *pFromContainer, char *pToContainer, int iHuser, int iHcat, int
iWriteMode = 0);

   /**
   MODIFY-ZIP-OPTIONS  *************************************
   It returns 0 if ok.
   char *pCrypto = crypto password
   int iLen      = length of crypto password
   int iEncrypt  = encryption or not
   **/
   int ModZipopt(char *pCrypto, int iLen, int iEncrypt);

   /**
   // reorganize the file by rewriting only the files having a header in the central
directory
   It returns 0 if ok.
   char *pContainer = name of the zip container file
   this file name must be valid
   if "link=linkname", the container is opened
   using this link name
   **/
   int ReorganizeZip(char *pContainer);
   /**
   Activate the trace processing.
   It returns 0 if ok.
   char *pFileName = name of the trace file.  If NULL, default name is used
   **/
   int ActivateTrace(char *pFileName);
   /**
   Deactivate the trace
   **/
   int DeactivateTrace();
   /**
   Check if trace is active
   **/
   bool IsTraceActivated();
   /**
   Get trace file name
   **/
```

```
   void GetTraceFilename(char *traceName);
   /**
   Return the zip container comments into the input buffer with the specified
   input size.  If this size is not sufficient, the data are truncated.
   char *Comments    = buffer address
   int iCommentsSize = buffer size
   **/
   int GetComments(char *Comments, int iCommentsSize);
   /**
   Allows to set or reset that K2 has been intercepted at interface level.
   bool b = indicates that K2 indicator must be set or reset
   **/
   void K2given(bool b);
   /**
   Activated if the 'Cpu Exhausted' event occurs.
   **/
   void CpuExhausted();
   /**
   Activated if the 'Term' event occurs.
   **/
   void Term();
   /**
   Global return code
   **/
   //void *pRcErr; // CRcErr m_RcErr;
   //#define RcErr (*((CRcErr*)pRcErr))
   /**
   Return the last error maincode.  Use for LIST problem when -1 is returned
   **/
   int GetLastError();
// private:
   /**
   Get the file name according to the link name.  The file name
   is returned in the ContainerName variable.
   ReadTFT rc is returned.
   char *pLink = link name
   **/
   long GetFileFromLink(char *pLink, char *pContainer);
   /**
   Check if the file exists.
   It returns 1 if the file exists, -1 if it exists but is empty,
   0 if it does not exist.
   char *pFileName = file name
   **/
   int FileExists(char* pFileName);
   /**
   Build a file name for output: :catid:$uid.FILEyyyymmdd.hhmmss
   **/
```

```
void BuildFileName(char *fn, int i, char *catid=0, char *userid=0);
/**
Validate if the input file name is a valid file name
Return true is bs2000 file name compliant.
const char *fn = file name
**/
bool IsBS2000FileName(const char *fn);
/**
Not implemented
**/
static void fsRout(const char* fn, int i);
/**
Convert input string in upper case
char *s = string to convert
**/
void ToUpperCase(char *s);
/**
Catalog a container (PAM, STD16)
char *cont = container name
char *link = link name
int format = container format
bool space = true (define space in prg)
**/
int CatalogContainer(char *cont, char *link, int format, bool space);
/**
Get new file name using SDF wildcard construction
char *selection
char *construction
char *srcname
char *newname
**/
int GetNewFileName(char *selection, char *construction, char *srcname, char *newname);
/**
Get new file name using SDF wildcard construction
char *selection
char *construction
void *src = fileitem object
char *newname
**/
int GetNewFileName(char *selection, char *construction, void *src, char *newname);
/**
Common function adding files or lib elements in ZIP container
char *pFileNames = files or lib file name
char *pElements = null ptr or element selector
char *pVersion = null ptr or version selector
char *pType = null ptr or type selector
char *pBase = null ptr or base version base selector
int iConvertMode = convert mode
```

```
   int iLevel       = compression level
   int iDataType    = data type (not specified, character, binary)
   char *pToFiles   = target construction file name
   int iTofileOption = specifies if libr element version and/or type are saved in the name
   int iDeleteOption = library element delete option                    132
   int iLogOption = logging option                                      133
   long nbr = number of added files                                     133
   **/
   int AddItems(char *pFileNames, char *pElements, char *pVersion, char *pType,
             char *pBase, int iConvertMode, int iLevel, int iDataType,
    char *pToFiles, int iTofileOption, int iDeleteOption, int iLogOption, unsigned long
&nbr, void *flist, bool pname);

   /* methods added for selection enhancement 1.2G */

   /* internal method
      pre-requisite CAddParam object has been created with all the parameters of the
      add statement
   */
   int AddToZip(void *paddparam);

   void SetExtractLogging(bool log);

   void SetSelectCCS(char *from, char *to);

   char    ContainerName[55];
   int     iCFormat;
   char FromCCS[9];
   char ToCCS[9];
   void *pZip;  //CZipArch m_Zip;
   #define Zip (*((CZipArch*)pZip))
   bool bRelLink;
   int m_OpenMode;
   bool k2Pressed;
   void UpdateUserInfo();
   bool bLib;
   int m_TofileOption;
   void *pAddParam; // CAddParam instance
   bool bExtractLoggingMax;                  // 134
   bool reorganizationNeeded; // 135
};
// extern bool glb_sysout;  // msg on sysout by default
#endif // __SzpZip_h__
```

5.2 SZPZOUT.H

This C++ header file describes the contents of the buffer returned by SHOW-FILE-ATTRI-BUTES operation (*ListFiles ()* function). The buffer returned by the function must be released by the caller.

```
#ifndef __SzpZout_h__
#define __SzpZout_h__
/*****************************************************************
 *  Classes        --
 *  File           szpzout.h
 *
 *  Copyright      (c) 'FUJITSU TECHNOLOGY SOLUTIONS' '2009'
 *
 *  Description    Output Find structure layouts
 *
 *  Version        120
 *  Date           2008-03-02
 *  Author         Ph. Dumont - OSL EPS
 *  Update         BS2ZIP v1.2A
 *
 *  Version        110
 *  Date           2004-09-30
 *  Author         L. Tambour - OSL EPS
 *  Update         BS2ZIP v1.1A
 *
 *  Version        100
 *  Date           2003-06-02
 *  Author         J. Beaume - OSL EPS
 *  Update         BS2ZIP v1.0B
 *****************************************************************/
struct szpOutSummary {
   unsigned int uRecSize;        // total size of a returned item
   void *pHeader;                // pointer to CFileHdr object
   unsigned short uFileNameSize; // file name size
   unsigned short BS2Flag;       // 1 = BS2000 file, 0 = others
   unsigned short ENCFlag;       // 1 = Encrypt. file, 0 = others (V120)
   // File name
};
struct szpOutFull {
   unsigned int uRecSize;        // total size of a returned item
   void *pHeader;                // pointer to CFileHdr object
   unsigned short uFileNameSize; // file name size
   unsigned short iBS2Flag;      // 1 = BS2000 file, 0 = others
   unsigned short iENCFlag;      // 1 = Encrypt. file, 0 = others (V120)
   char uModTime[8];             // last mod file time
   char uModDate[10];            // last mod file date
```

```
   //unsigned short uFiller;       // unused   V120
   long long uComprSize;         // compressed size
   long long uUncomprSize;       // uncompressed size
   unsigned short uCommentSize;  // file comment size
   unsigned short uExtraFieldSize; // extra field length
   unsigned int uFiller2;         // unused
   // Filename
   // Extrafield
   // Comment
};
#endif // __SzpZout_h__
```

Output buffers examples

1. INFORMATION=*SUMMARY

szpOutSummary structure	File name 1	szpOutSummary structure	File name 2
SzpOutSummary structure	File name 3		

1. INFORMATION=*ALL

szpOutFull structure	File name 1	Extra Fields 1	Comments 1
szpOutFull structure	File name 3	Comments 2	

5.3 BS2ZIPPR LLM

The run time BS2ZIPPR must be linked with the TU application that wants to use BS2ZIP as a sub-program.

5.4 Program example

```
#include "tstzip.h"
#include "SzpZout.h"
#include <string.h>
#include <stdio.h>
void main() {
   // create a CSzpZip object with error reported in rc and log file.
   CSzpZip *zip = new CSzpZip(false);
   // create a new container (BS2000 format by default)
   int rc = zip->OpenZip("MYCONT.ZIP", CSzpZip::updateNew, CSzpZip::defaut);
   // add a file to the container
   rc = zip->AddFiles("MYFILE.TXT",
                          CSzpZip::std,
                          CSzpZip::defaultCompression,
                                                  CSzpZip::character,
                                                  "MYFILE.TXT"
                                                  );
   // list the contents of the container
   char *pBuf = 0;
   int iSize = 0;
   int iNumber = 0;
   // Get first element and loop while rc = 1;
   rc = zip->ListFiles("*", &pBuf, &iSize, CSzpZip::infoSummary, true, false);
   while(rc == 1) {
          iNumber++;
          // do something with buffer
          ...
          delete [] pBuf;
          rc = zip->ListFiles("*", &pBuf, &iSize, CSzpZip::infoSummary, true, false);
   }
   printf("Number of matching files = %d\n", iNumber);
   // extract file from the container – extracted file = EXT-MYFILE.TXT
   rc = zip->ExtractFiles("*", "EXT-*", CSzpZip::any, CSzpZip::notSpecified,
                          CSzpZip::std, CSzpZip::keep, false);
   // close zip container
   rc = zip->CloseZip();
   delete zip;
   }
```

6 Related publications

You will find the manuals on the internet at http://bs2manuals.ts.fujitsu.com.

[1] **SDF** (BS2000)
SDF Dialog Interface
User Guide

[2] **BS2000 OS DX**
Commands
User Guide

[3] **JV** (BS2000)
Job Variables
User Guide

[4] **BS2000 OS DX**
DMS Macros
User Guide

[5] **XHCS** (BS2000)

8-Bit Code and Unicode Processing in BS2000
User Guide